7-2020

Multi-Homotopy Class Optimal Path Planning for Manipulation with One Degree of Redundancy

Jacob J. Rice
Marquette University

Joseph M. Schimmels
Marquette University, joseph.schimmels@marquette.edu

Mechanism and Machine Theory, Vol. 149 (July 2020): 103834. DOI.

Multi-Homotopy Class Optimal Path Planning for Manipulation with One Degree of Redundancy

Jacob J. Rice
Department of Mechanical Engineering, Marquette University, Milwaukee, WI
Joseph M. Schimmels
Department of Mechanical Engineering, Marquette University, Milwaukee, WI

Abstract

Redundant manipulators have an infinitely large set of joint paths that yield a desired end-effector path in the task space. A unique joint path can be obtained by minimizing a global cost function. Prior optimal control methods minimize a global cost function to find a local minimum within a homotopy class. Many possible locally optimal joint paths are in different homotopy classes. This paper presents an algorithm that effectively searches the solution space and finds many locally optimal paths in all relevant homotopy classes. The path with the lowest cost is very likely the globally optimal path. The algorithm is demonstrated in a case study for which the globally optimal path would be impossible to find using traditional methods.

Keywords
Redundant robots, Path planning for manipulators, Homotopy class, Roadmap, Kinematics

1. Introduction
This paper considers redundant robot manipulation in which a specified *task path* (described by *m* coordinates) is realized by commanding a *joint path* (described by *n* coordinates), for which the degree of redundancy is $r = n - m = 1$.

The problem of finding a specific joint path that realizes the task path is called the redundant inverse kinematic (RIK) path planning problem and is commonly resolved using optimization. The joint path may be resolved *instantaneously* (often referred to as local resolution) by identifying the optimal joint velocity for advancing the task [1]. Starting from an initial joint configuration, the joint path is then generated by integrating the instantaneous solution over the task path. Alternatively, the joint path may be resolved *globally* by identifying the joint path that minimizes a *global cost function* (an integral cost criterion) while satisfying the task path and appropriate boundary conditions.

Global resolution of the RIK path planning problem has been studied since the 1980's [2], [3], [4], [5], [6], [7]. Unfortunately, the terms used to describe the method of RIK resolution conflict with terms used to describe conventional optimization. Prior work on "global optimization" presented *global resolution* methods for finding a *local* minimum of the global cost function. These methods, in general, do not yield the global minimum. The existence of multiple locally optimal paths for the RIK path planning problem has been demonstrated in case studies [5], in which multiple locally optimal paths are shown to exist in different homotopy classes. Joint paths in different homotopy classes cannot be continuously deformed into each other without violating the task path or the boundary conditions. The existence of multiple homotopy classes complicates the problem of identifying the joint path that globally minimizes the global cost function. The existence of multiple locally optimal paths in the same homotopy class further complicates the problem.

1.1. Optimal paths and homotopy classes
Locally optimal joint paths depend on the RIK solution space and the selected boundary conditions. The *RIK solution space* is the admissible space of joint paths and is described by a time-indexed sequence of self-motion manifolds [8], which are sets (or groups of sets) of connected joint configurations that yield the same end-effector configuration. A simplified RIK solution space for $r = 1$ is illustrated by the shaded surface in Fig. 1, where dotted contour lines show the surface topography, dashed lines are self-motion manifolds (which are paths for $r = 1$) at various task instances, and solid lines with circle endpoints are alternative joint paths.

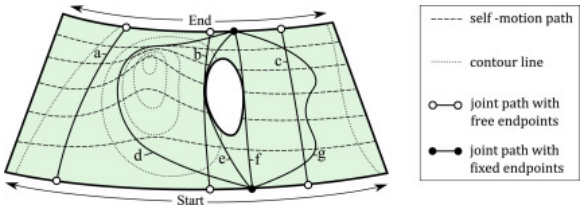

Fig. 1. An RIK solution space with two homotopy classes separated by a single hole.

The joint path endpoints must be on the self-motion manifolds of the task path endpoints, regardless of the boundary conditions. For fixed boundary conditions, the endpoint joint configurations are user-selected based on some other criterion and the optimization is constrained by them. For free (unconstrained) boundary conditions, the endpoint joint configurations are determined by the optimization. In general, locally optimal paths with free boundary conditions have lower global cost values than those with fixed boundary conditions.

Paths a–f in Fig. 1 are paths that locally minimize the global cost function (corresponding to the length of the joint path), whereas path g is not locally optimal. The paths with free boundary conditions are typically shorter than those with fixed boundary conditions (a, b, and c are shorter than d, e, and f, respectively).

The number of locally optimal joint paths depends on the connectivity of the RIK solution space and on its nonlinearities with respect to the global cost function. Structural features impacting the connectivity of the RIK solution space, such as the surface hole (a set of joint configurations that are unreachable due to joint limits, task-space obstacles, or self-collisions) in Fig. 1, induce bifurcations in the self-motion manifolds. A progressing joint path must take one of two branches (each side of the hole). Paths that take different branches cannot be continuously deformed into each other. They are in different homotopy classes. The number of homotopy classes increases with the number of bifurcations. This paper focuses on bifurcations induced by *kinematic singularities*, joint configurations where arbitrary task motion is not possible. Bifurcation kinematic singularities correspond to saddle points on the RIK solution space, where a joint path must travel on one side of the saddle or the other.

When a large number of homotopy classes exist, it is highly unlikely that the globally optimal path will be found without a multi-search strategy. Moreover, the obtained joint path may not be the best path even in its own homotopy class.

Each homotopy class may have multiple locally optimal paths due to cost function nonlinearities in the RIK solution space. This is illustrated in Fig. 1, where the hill in the surface induces locally optimal paths on opposite sides of the hill. Unlike the paths separated by the hole, joint paths on either side of the hill (e.g., paths a and b in Fig. 1) can be deformed into each other along the self-motions.

Three levels of "optimal paths" are described in this paper and illustrated by the joint paths in Fig. 1:

1. A *locally optimal path* minimizes the global cost function such that any infinitesimal deformation of the path yields a higher global cost value. It is the best path in a neighborhood of paths (e.g., paths a–c for free boundary conditions and paths d–f for fixed boundary conditions).
2. A *homotopy optimal path* is the locally optimal path with the lowest global cost of all locally optimal paths in its homotopy class. It is the best path in its homotopy class (e.g., paths a and c for free boundary conditions and paths e and f for fixed boundary conditions).
3. A *globally optimal path* is the homotopy optimal path with the lowest global cost of all homotopy optimal paths. It it the best path in the entire optimization domain (e.g., path c for free boundary conditions and path f for fixed boundary conditions).

The globally optimal path with free boundary conditions is the best path possible. Prior work in RIK path planning has not identified systematic procedures for identifying the best path when multiple homotopy classes exist or when multiple locally optimal paths exist in a homotopy class.

1.2. Prior work in global resolution

Global resolution of the RIK path planning problem is usually framed as an optimal control problem which may be solved "indirectly" [2], [4], [5], "directly" [6], [7], or using dynamic programming [9]. Dynamic programming requires a fixed boundary condition and converges to the constrained globally optimal path, which is sub-optimal compared to the globally optimal path with free boundary conditions. This paper focuses on solving the more difficult problem of finding the globally optimal path with free boundary conditions.

The optimal control problem, for any type of boundary condition, is solved "indirectly" by seeking to satisfy necessary conditions for optimality given by Pontryagin's maximum principle [2] or the Euler-Lagrange equation [4], [5]. If the optimal joint path does not encounter a solution space hole (e.g., joint limit), it is the

solution to a two-point-boundary-value problem. Shooting methods are frequently used to solve these problems, but often suffer from numerical instability when the Euler-Lagrange equation is stiff [10].

A more robust way to solve the optimal control problem, for any type of boundary condition, is to solve it "directly" using nonlinear programming [11] where the joint path is represented by a finite number of parameters (e.g., approximating the joint path with a spline curve with a finite number of nodes [7]). The joint path is iteratively improved by descending the gradient of the cost function with respect to the path parameters. This solution method requires an initial joint path and iteratively deforms the path over the RIK solution space into a locally optimal joint path.

The likelihood of an optimal control solver finding the globally optimal path depends on the complexity of the RIK solution space. The number of joint path homotopy classes provides some measure of the solution space complexity. None of these existing approaches first identifies the set of homotopy classes in which the globally optimal path may exist.

1.3. Prior work in homotopy class identification

Most work in homotopy class identification has addressed tasks without path constraints and with fixed endpoints (e.g., mobile robot path planning around obstacles) [12], [13], [14], [15], [16], [17], [18], [19], [20]. The set of homotopy classes can be described by finding a single path in each homotopy class. The network of paths is captured by a *roadmap*, a graph with nodes corresponding to manipulator configurations and edges corresponding to feasible joint paths generated with an instantaneous path planner (often referred to as a local path planner). The discrete representation of a joint path is given by a *route*, a sequence of edges connecting nodes, where the start and terminal nodes are configurations corresponding to the start and end of the task.

Roadmap nodes are commonly generated by sampling the configuration space (e.g., joint space) [21]. These nodes, in general, do not satisfy the task path constraints, but nearby configurations on the task path can be found using Jacobian-based inverse kinematic methods [22], [23], or using rapidly-exploring-random-trees (RRTs) [23], [24], [25], [26]. Note, RRTs use a local planner to build the roadmap outward from existing nodes, whereas traditional roadmaps use a local planner to find edges between existing nodes. The local planner is usually a linear motion in the joint space that lifts off the nonlinear RIK solution surface resulting in task error between the nodes. Therefore, the nodes must be close together to limit task error, resulting in large roadmaps.

In general, sample-based roadmaps do not have the property of having a unique route for each homotopy class. A discrete homotopy relation is used in [12], to simplify a sampled roadmap to have this property. However, this method is only applicable for 2D Cartesian environments with obstacles. Roadmaps with this homotopy capturing property can be generated by using obstacle information to create a Voronoi graph [13], [14]. Again, this method is limited to Cartesian spaces with obstacles.

The set of homotopy classes can also be described by identifying homotopy invariants [15], [16], [17], [18], [19], [20]. For example, for mobile robots traveling point-to-point in a plane with obstacles, the homotopy class of the path can be identified by a "word" corresponding to the path's sequence of crossing non-intersecting rays emanating from the obstacles [18] (similar concepts are used in [16], [17]). Homotopy invariants are also described for mobile robot problems in 3 and 4 dimensional Cartesian spaces [18].

1.4. Approach

This paper presents a 5-step algorithm for finding the globally optimal joint path satisfying a specified task path for manipulation with one degree of redundancy. Inequality constraints from task-space obstacles, joint limits, or self-collisions are not considered. The process involves: 1) strategically decomposing the task into a set of sub-tasks, 2) finding multiple sub-optimal paths for each sub-task, 3) deforming these sub-optimal paths into

multiple locally optimal solutions of the sub-tasks, 4) strategically concatenating sub-task solutions to obtain sub-optimal complete solutions, and 5) deforming these sub-optimal complete solutions into locally optimal complete solutions. Steps 3 and 4 are needed to address the problem of multiple locally optimal paths existing in the same homotopy class. The algorithm yields a set of many locally optimal paths that are continuous, smooth, and accurately track the equality constraints of the task path. The locally optimal path within this set having the lowest global cost is very likely, but is not guaranteed, to be the globally optimal path.

The algorithm decomposes the task into sub-tasks by dividing the complete task path at instances when self-motion paths bifurcate (or converge). Each sub-task RIK solution space has a relatively simple connection structure and is bounded by self-motion paths associated with the bifurcation points or with the complete task endpoints. Knowledge of the connectivity of these sub-task RIK solution spaces is necessary because they relate to joint path homotopy classes. This connectivity structure is characterized by a new directed graph called the bifurcation branch roadmap (bb-roadmap). A simple open manifold example is illustrated in Fig. 2. Each node of the bb-roadmap is a self-motion path of a sub-task endpoint (a self-motion path of either a bifurcation point or a complete task endpoint). Each edge of the bb-roadmap is a *sub-task homotopy class* (a homotopy class of joint paths that satisfy the sub-task). Each route through the bb-roadmap (sequence of edges connecting endpoint nodes) corresponds to a (complete task) homotopy class and every relevant homotopy class has a corresponding route. The bb-roadmap construction requires an abstract characterization of how bifurcations of the self-motion manifolds impact the joint path homotopy classes. The abstract characterization depends on the number and structure[1] of the self-motion manifolds over the task path. With this abstract characterization, the bb-roadmap is quickly generated with little computation. The bb-roadmap is the output of the first step of the algorithm and is used by the subsequent steps.

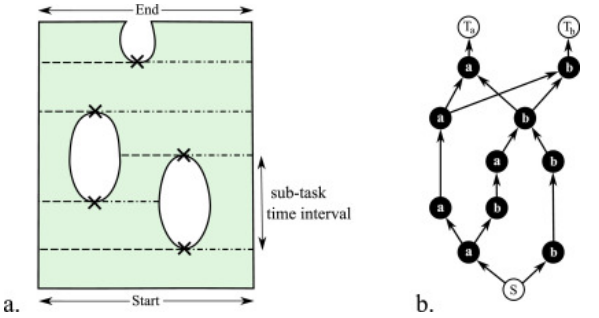

Fig. 2. An RIK solution space and the bifurcation branch roadmap (bb-roadmap) characterizing its connection structure. a) RIK solution surface with self-motion in the horizontal direction and task progress in the vertical direction. b) Corresponding bb-roadmap. Each self-motion path touching a bifurcation point ($\times$) is split into two separate paths (dashed and dashed-dotted lines). These paths correspond to bb-nodes (black circles). Self-motion paths of the complete-task endpoints correspond to endpoint bb-nodes (white circles). Homotopy classes of joint paths connecting sub-task endpoint self-motion paths correspond to directed edges (arrows).

The remaining steps of the algorithm alternate between using a path planner based on instantaneous RIK resolution to obtain sub-optimal paths within desired sub-task homotopy classes and using a “direct” global RIK resolution method to deform the sub-optimal paths into locally optimal paths. The final step, in which complete paths are deformed into locally optimal paths, requires the most computation time, especially when there are many homotopy classes to investigate. The bifurcation branch algorithm uses an upper/lower bound method to eliminate a large number of homotopy classes that cannot contain the globally optimal path. For problems with many bifurcation points, the set of homotopy classes considered in the final step is reduced by a factor of 100 or more.

1.5. Paper contributions and overview

This paper presents an algorithm called the bifurcation branch algorithm (bb-algorithm) for identifying the globally optimal path for the RIK path planning problem with one degree of redundancy involving multiple homotopy classes. The primary contributions of this paper relate to the introduction of the bifurcation branch roadmap (bb-roadmap), an abstract characterization of how bifurcations of one or two closed self-motion manifolds impact the joint path homotopy classes, and a procedure for rapidly constructing the bb-roadmap for solution spaces with bifurcations from kinematic singularities. The secondary contributions include an instantaneous path planner for generating initial paths with specified endpoints and a robust path deformation method for deforming a sub-optimal path into a locally optimal path. The effectiveness of the bb-algorithm is demonstrated on a manipulation problem with a 3R manipulator tracing a path in a 2D task space with its end-effector. Although this type of problem is often assumed to be very simple, this paper demonstrates that the complexity depends on the task path for which the solution space may have multiple bifurcations yielding multiple homotopy classes (e.g., one case study in this paper has 4,556,250 different homotopy classes).

The paper is organized as follows: Section 2 reviews relevant background of the RIK path planning problem, Section 3 describes the bb-algorithm in greater detail, Section 4 describes the bifurcation branch roadmap (bb-roadmap) construction, Section 5 describes the instantaneous path planner, Section 6 describes the path deformation procedure, Section 7 demonstrates the algorithm for a case study in which the number of homotopy classes is *very* high and the globally optimal path cannot be found using traditional methods, and Section 8 summarizes the results.

2. Technical background

This section reviews the technical background and terminology associated with the redundant inverse kinematic (RIK) path planning problem and prior approaches to its resolution.

2.1. Terminology

Path - A path in a topological space A is a continuous map $\mathbf{a}(\rho)\colon I \to A$,, where I is the unit interval [0,1], $\rho \in I$ is the indexing parameter, $\mathbf{a}(0)$ is the start point, and $\mathbf{a}(1)$ is the terminal point.

Three types of paths are used in this paper:

1. a *task path* $\mathbf{x}(t)\colon I \to X$,
2. a *joint path* $\mathbf{q}(t)\colon I \to Q$, and
3. a *self-motion path* $\mathbf{q}(\psi)\colon I \to Q$.

Task paths and joint paths are both parameterized by normalized time *t* denoting task progress, but are in different topological spaces. Joint paths and self-motion paths are in the same topological space Q, but are parameterized differently. A self-motion path corresponds to a connected set of joint configurations yielding the same end-effector configuration in task space. The self-motion parameter ψ is orthogonal to t.

Path homotopy - A path-homotopy is a continuous mapping $H(t,\psi)\colon I \times I \to Q$,, such that $H(t,0) = \mathbf{q}_0(t)$ and $H(t,1) = \mathbf{q}_1(t)$,, where $\mathbf{q}_0(t), \mathbf{q}_1(t)\colon I \to Q$ are arbitrary paths with the same endpoints.[2]

A path homotopy describes the deformation of the joint path along self-motion paths.

Homotopy Class (of a joint path) - The homotopy class of a joint path $\mathbf{q}(t)$ is the set of all joint paths for which a path homotopy to $\mathbf{q}(t)$ exists. In less precise language, a homotopy class is the collection of all joint paths that can be deformed into each other.

2.2. Redundant inverse kinematics

Consider a manipulator with n joints in which the end-effector performs a task described by m coordinates, where the degree of redundancy is $r = n - m$. The forward kinematic map $\mathbf{f}(\mathbf{q})$: $Q \to X$ is a nonlinear function that defines the relationship between the manipulator's configuration in joint space $\mathbf{q} \in Q$ (e.g., relative angles between consecutive links) and the end-effector configuration in task space $\mathbf{x} \in X$ (e.g., position and orientation of the end-effector frame relative to the base frame). A unique inverse map does not exist when $r > 0$.

The RIK path planning problem is to find a joint path $\mathbf{q}(t) \in Q$ such that $\mathbf{f}(\mathbf{q}(t)) = \mathbf{x}(t)$. The RIK path planning problem does not have a unique solution, but a unique path can be obtained using instantaneous or global resolution.

2.3. Instantaneous RIK resolution

A specific joint path can be obtained by starting at a joint configuration, $\mathbf{q}_0 \mid \mathbf{f}(\mathbf{q}_0) = \mathbf{x}(0)$,, and integrating the following instantaneous (velocity-based) optimization:

(1)

$$\begin{aligned} \text{min.} \quad & \frac{1}{2}(\dot{\mathbf{q}} - \dot{\mathbf{q}}_N)^T \mathbf{W}(\dot{\mathbf{q}} - \dot{\mathbf{q}}) \\ \text{s.t.} \quad & \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} = \dot{\mathbf{x}}, \end{aligned}$$

where $\dot{\mathbf{q}}$ and $\dot{\mathbf{x}}$ are instantaneous motions in the joint and task spaces, respectively, $\mathbf{J}(\mathbf{q}) = \partial \mathbf{f}(\mathbf{q})/\partial \mathbf{q}$ is the Jacobian matrix, $\dot{\mathbf{q}}_N$ is the "preferred" joint motion [27], and **W** is a positive definite matrix that defines the distance metric in the joint tangent space.

The joint motion $\dot{\mathbf{q}}$ that solves (1) is readily identified [28] using:

(2)

$$\dot{\mathbf{q}} = \mathbf{J}^{\dagger}\dot{\mathbf{x}} + (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})\dot{\mathbf{q}}_N,$$

where $\mathbf{I}$ is the identity matrix and $\mathbf{J}^{\dagger}$ is the weighted pseudoinverse calculated by

(3)

$$\mathbf{J}^{\dagger} = \mathbf{W}^{-1}\mathbf{J}^T(\mathbf{J}\mathbf{W}^{-1}\mathbf{J}^T)^{-1}$$

when $\mathbf{J}$ is full-rank rank($\mathbf{J}$) $= m$.

The first term on the right side of (2) yields the weighted minimum norm joint motion that produces the desired task motion $\dot{\mathbf{x}}$. The second term is a joint self-motion that causes no end-effector task motion; it is usually set to zero $\dot{\mathbf{q}}_N = \mathbf{0}$.

2.4. Global RIK resolution

A unique joint path can alternatively be obtained using global optimization by solving:

(4)

$$\begin{aligned} \text{min.} \quad & G_{\text{global}} = \int \frac{1}{2}\dot{\mathbf{q}}(t)^T \mathbf{W}\dot{\mathbf{q}}(t)dt, \\ \text{s.t.} \quad & \mathbf{f}(\mathbf{q}(t)) = \mathbf{x}(t) \end{aligned}$$

where G_{global} is the global path cost.

A locally optimal joint path can be found using "direct" methods by deforming an initial joint path (a required input). An initial joint path can be obtained using instantaneous RIK resolution. The number of locally optimal joint paths obtained using global resolution depends on the RIK solution space structure.

2.5. RIK solution space structure

The RIK solution space is regarded as a continuous sequence of preimages of each $\mathbf{x} \in \mathbf{x}(t)$ under $\mathbf{f}$. Each preimage (set of all inverse kinematic solutions) of a task configuration is characterized by a set of disjoint *r*-dimensional self-motion manifolds:

(5)

$$\mathbf{f}^{-1}(\mathbf{x}) = \bigcup_{i}^{n_s} \boldsymbol{\mathcal{M}}_i(\boldsymbol{\psi})$$

where n_s is the number of disjoint self-motion manifolds $\boldsymbol{\mathcal{M}}_i(\boldsymbol{\psi})$,, and $\boldsymbol{\psi}$ is a set of independent parameters $\boldsymbol{\psi} = \{\psi_1, \psi_2, \dots, \psi_r\}$ (*r* is the degree of redundancy). Each self-motion manifold is a set of all the joint configurations that yield the same task configuration that are continuously connected. When $r = 1$, the self-motion manifold can be represented by a path in joint space $\mathbf{q}(\psi)$: $I \to Q$ parametrized by ψ orthogonal to t.

The connectivity of self-motion manifolds is investigated in [8] by identifying homotopy classes *of self-motion manifolds*. The workspace of the manipulator is divided into $W-$ sheets, such that the self-motion manifolds of all task configurations within a $W-$ sheet are in the same homotopy class (i.e., the self-motion manifolds can be continuously deformed into each other). Each $W-$ sheet is a connected set of regular values bounded by coregular values or critical values [8], where:

- a *critical value* is a *task* configuration $\mathbf{x}_s$ for which $\mathbf{f}^{-1}(\mathbf{x}_s)$ is a kinematic singularity,
- a *kinematic singularity* is a *joint* configuration $\mathbf{q}_s$ where the Jacobian matrix is rank deficient (i.e., rank($\mathbf{J}(\mathbf{q}_s)) < m$),
- a *regular value* is a *task* configuration $\mathbf{x}$ for which $\mathbf{f}^{-1}(\mathbf{x})$ does not contain a singularity, and
- a *coregular value* is a *task* configuration $\mathbf{x}_c$ for which $\mathbf{f}^{-1}(\mathbf{x}_c)$ is a *coregular self-motion path*[3] which contains both singular and non-singular joint configurations.

Critical values $\mathbf{x}_s$ exist at the workspace boundaries, whereas coregular values $\mathbf{x}_c$ exist at the interface between neighboring $W-$ sheets,. The RIK path planning problem is much simpler when the task is inside a single $W-$ sheet for which the number of homotopy classes is equal to the number of self-motion manifolds of the $W-$ sheet. For task paths that cross coregular values, the self-motion manifolds do not deform continuously over the task, but bifurcate, resulting in a more complicated RIK solution space structure with multiple locally optimal paths.

3. Bifurcation branch algorithm

The existence of many locally optimal solutions to the RIK path planning problem greatly increases the difficulty of obtaining the globally optimal path. This section describes the bifurcation branch algorithm (bb-algorithm) that overcomes this difficulty for RIK problems with one degree of redundancy.

The algorithm is referred to as the bifurcation branch algorithm because it strategically uses the self-motion paths (branches) associated with the bifurcation points to effectively search the RIK solution space. The algorithm finds the globally optimal path (to a high degree of certainty) with 5 steps:

Step 1: Solution Space Structure Characterization
Step 2: Initial Path Network Generation
Step 3: Path Segment Deformation
Step 4: Refined Path Network Generation
Step 5: Complete Path Deformation

using 3 procedures:

Procedure 1: Bifurcation Branch Roadmap Construction
Procedure 2: Bi-directional Instantaneous Path Generation
Procedure 3: Path Deformation.

The steps are illustrated in Fig. 3 on a simple RIK solution space with open and bounded self-motion manifolds in the horizontal direction and task progress in the vertical direction. The following subsections describe the *steps*, how the procedures are used in them, and why these steps are important for finding the globally optimal path. Details of the *procedures* are provided in the following Sections. Procedure 1 is different for RIK solution space structures with different numbers and structures of self-motion manifolds. The general details of Procedure 1 are included in the discussion of Step 1 along with specific details for the RIK structure in Fig. 3.

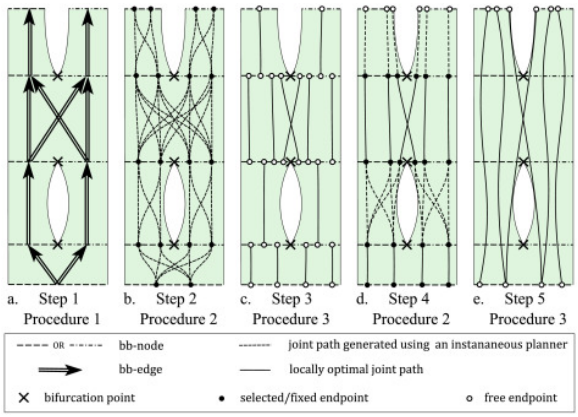

Fig. 3. The states of the bifurcation branch algorithm, shown on a simplified abstract RIK solution space, after each step using a procedure. Steps: 1) characterize the solution space with the bb-roadmap, 2) generate a network of initial joint paths. 3) deform the initial path sub-task segments into locally optimal paths with free boundary conditions, 4) join locally optimal paths together into a refined network of joint paths. 5) deform the complete paths into locally optimal paths.

3.1. Step 1: solution space structure characterization

The RIK solution space is characterized with respect to homotopy classes by constructing the bifurcation branch roadmap (bb-roadmap). The process for constructing the bb-roadmap involves three sub-steps.

The first sub-step decomposes the task into sub-tasks by dividing the task path at instances when self-motion manifolds bifurcate or converge (at each $\times$). A means of identifying these bifurcation points is required. Bifurcation points corresponding to kinematic singularities have geometric conditions associated with them. Similarly, the corresponding coregular values also have geometric conditions. These conditions have already

been identified for some manipulator structures including n-R planar serial manipulators [29], a non-planar 4R manipulator [8] and a 7R spatial manipulator [30]. These geometric conditions, if known, are used with root finding over the task path to quickly identify all coregular values along the specified task path. The result of the first sub-step is the identification of the time instances of the bb-nodes (nodes of the bb-roadmap).

The second sub-step generates numerical descriptions (e.g., spline curves) of the self-motion paths associated with each bb-node. Each bb-node is described by a single self-motion path of either a complete task endpoint or a coregular value. A complete task endpoint may have multiple self-motion manifolds and therefore multiple bb-nodes. The self-motion path of a coregular value is called a coregular self-motion path. It is not strictly a manifold because it intersects a singularity. The coregular self-motion path is divided at the singularity into two separate self-motion paths called bifurcation branches. Each bifurcation branch corresponds to a bb-node. Joint paths that cross different sequences of bb-nodes are in different homotopy classes. The possible bb-node sequences are defined by the roadmap edges.

The third sub-step identifies the edges of the bb-roadmap connecting the bb-nodes. Edges can be implicitly identified based on prior abstract characterization of how bifurcations in the self-motion manifolds impact the number of joint path homotopy classes. The abstract characterization depends on the number *and* structure of the self-motion manifolds.

The abstract characterization is trivial for problems with *open* manifolds in which a joint path homotopy class splits in two when its self-motion manifold splits in two. This characterization yields a single joint path homotopy class (a single bb-edge) connecting a pair of bb-nodes. For the open manifold example in Fig. 3, each bb-edge (double lined arrows) corresponds to a combination of bb-nodes (a start node and a terminal node in the same sub-task) that can be connected by a joint path. Whether or not the edge exists depends directly on the number of self-motion manifolds inside the sub-task.

The RIK solution space structure for *closed* self-motion manifolds is much more complex. There can be *multiple* bb-edges connecting the same pair of bb-nodes (as shown in Section 4 for one or two closed self-motion manifolds).

The identification of the number and structure of the self-motion manifolds can be accomplished by 1) looking up this information if the $W-$ sheets, have been previously characterized [8], 2) checking geometric conditions (e.g., [29] for planar manipulators), or 3) numerically generating all self-motion paths for a single task point in each sub-task. All self-motion paths of a single task point can be determined by sampling the joint space, then using a Jacobian-based inverse kinematic method to find inverse kinematic solutions for the desired task point, and finally using a self-motion path planner (described in Section 4.4) to connect these inverse kinematic solutions.

To summarize Step 1, the bb-roadmap takes advantage of high-level knowledge of the manipulator and is generated with very little computation. The bifurcation points are quickly and deterministically identified using known geometric conditions for kinematic singularities and coregular values of the manipulator. The bb-nodes and bb-edges are quickly identified from an investigation of the number and structure of self-motion manifolds in each sub-task. The result is a directed graph in which each route identifies a homotopy class of complete joint paths. Continuous self-motion paths associated with each bb-node are generated with a self-motion path planner. These self-motion paths define the structure of the RIK solution space and are used for effectively searching the entire solution space for the globally optimal path.

3.2. Step 2: initial path network generation

Initial, sub-optimal joint paths are generated in each sub-task homotopy class using instantaneous RIK resolution. Conventional instantaneous resolution does not control the terminal endpoint joint configuration

and therefore cannot control the homotopy class of the joint path. The bb-algorithm uses a new instantaneous path planner (described in Section 5) that bi-directionally generates a joint path between two specified endpoint joint configurations to control the homotopy class of the path. The endpoint joint configurations are sampled from the self-motion paths of the associated bb-nodes. The result of Step 2 is a *configuration roadmap* superimposed on the bb-roadmap, where each node is now a specific joint configuration and each edge is a specific joint path connecting sub-task start and end nodes.

Since each sub-task homotopy class can have multiple locally optimal paths, multiple sub-optimal paths are generated by connecting different combinations of sampled node configurations. Fig. 3b shows each bb-node with two sampled configurations (solid circles), and four initial joint paths (solid lines) in each sub-task homotopy class. Actual implementation uses three or more sampled joint configurations, equally spaced over the self-motion path, with two of the samples very close to the self-motion path bounds (i.e., near bifurcation points).

3.3. Step 3: path segment deformation

The instantaneous resolution generated sub-optimal joint paths in the sub-task homotopy classes (path segments obtained in Step 2) are deformed into locally optimal joint paths with free boundary conditions (solid lines with open circle endpoints in Fig. 3c). Because the RIK solution subspace corresponding to the sub-task homotopy class has relatively simple structure, with enough sampled points in Step 2, it is very likely[4] that *all* the locally optimal joint paths of sub-tasks are found. The best path among all locally optimal paths in the same sub-task homotopy class is the *sub-task homotopy optimal path*. However, all of these locally optimal paths are important because they capture different regions of the RIK solution space of lower global cost. The globally optimal path of the full task path will likely pass near some of these regions. This step is important for effectively searching the RIK solution space.

The locally optimal paths of adjoining sub-task homotopy classes do not have the same endpoint configurations and therefore cannot be combined into a continuous joint path, as shown by the discontinuous network of paths in Fig. 3c. A lower bound cost for the homotopy optimal path of each (complete task) homotopy class is obtained by summing the costs of the corresponding sequence of sub-task homotopy optimal paths.

3.4. Step 4: refined path network generation

The locally optimal paths of *every other* sub-task homotopy class in the sequence of sub-tasks are joined together, again using the instantaneous path planning procedure. These paths are illustrated as dashed lines in Fig. 3d. The result is a refined *configuration* roadmap, in which each route through the roadmap is continuous, smooth, and satisfies the task equality constraints for the complete task.

3.5. Step 5: complete path deformation

The sub-optimal joint paths of the refined configuration roadmap routes are deformed into locally optimal *complete* joint paths. The best of these paths is very likely the globally optimal joint path. Starting with the joint path from Step 4 with the lowest cost, this joint path is deformed into a locally optimal joint path. The cost of this locally optimal path is then used as an upper bound cost for the globally optimal path. Any homotopy class with a lower bound cost (computed in Step 3) greater than the upper bound cost cannot contain the globally optimal path and is removed from the set of homotopy classes considered in this step. Only paths in promising homotopy classes are deformed into locally optimal paths. For tasks with many bifurcations, the set of promising homotopy classes considered is reduced by a factor of 100 or more, saving computation time associated with the path deformation procedure.

A “direct” optimal control method (based on nonlinear programming) is used to deform each sub-optimal path into a locally optimal path. The bb-algorithm uses a new “direct” method that reformulates the global resolution

problem (4) into a reduced-order problem using self-motion paths. The modified direct method used in the bb-algorithm ensures the path deformation *reliably* converges to a locally optimal path within the same homotopy class and does not "jump" to a different homotopy class, which is of great concern, especially for sub-optimal paths passing near bifurcation points.

4. Bifurcation branch roadmap construction

As a manipulator executes a task path that crosses coregular values, the manipulator's self-motion manifolds bifurcate (split or join) at the coregular values. Bifurcation of the self-motion manifolds impact the number of joint path homotopy classes, but the impact can be quite different depending on the number and structure of the self-motion manifolds.

An abstract characterization of the impact that self-motion manifold bifurcations have on the number of joint path homotopy classes precedes the construction of the bifurcation branch roadmap. The abstract characterization for one or two *open* self-motion manifolds (like the RIK case illustrated in Fig. 2) is intuitive and results in bb-roadmaps with relatively simple structures. This section presents the more complex abstract characterization for manipulators with one or two *closed* self-motion manifolds. The correct characterization is not immediately intuitive and yields much more complex bb-roadmap structures. A method for quickly constructing the bb-roadmap for a specific task path that crosses many coregular values is introduced.

4.1. Tasks crossing a single coregular value

Consider a task path crossing a single coregular value, starting at a task configuration with one self-motion manifold, $\mathbf{q}_S(\psi)$,, and terminating at a task configuration with two self-motion manifolds, $\mathbf{q}_{T_a}(\psi)$ and $\mathbf{q}_{T_b}(\psi)$. The RIK solution surface of the task is illustrated in Fig. 4, where each solid line corresponds to a self-motion manifold (a connected set of joint configurations that yield the desired end-effector position at a specified time). As the task progresses, the self-motion manifold experiences a structural change at time t_c when the task path crosses the coregular value. The solution space structural change is due to a coregular self-motion path that is self-intersecting, shown by the dashed and dashed-dotted lines forming a figure-eight. The intersection point is a kinematic singularity, at which the Jacobian matrix is rank deficient and the dimension of the null space increases.

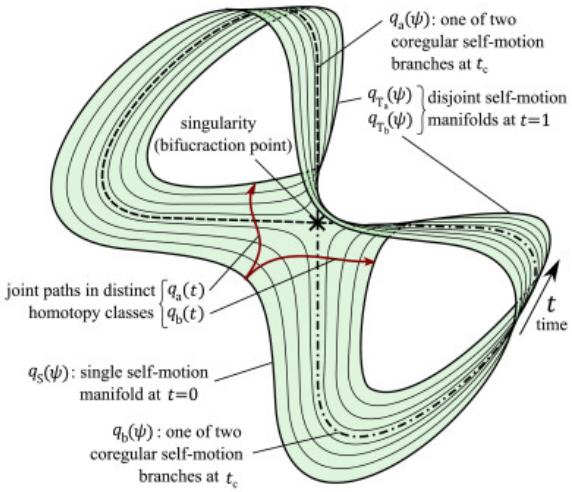

Fig. 4. RIK solution space of a task path crossing a coregular value at time t_c. Solid paths are self-motion manifolds at different time instances. The coregular self-motion path intersects itself at the kinematic singularity; it is separated into a dashed line path $\mathbf{q}_a(\psi)$ and dashed-dotted line path $\mathbf{q}_b(\psi)$. The 2 heavier solid lines with arrows are joint paths $\mathbf{q}_a(t)$ and $\mathbf{q}_b(t)$; they cannot be continuously deformed over the surface into each other.

A singularity associated with a coregular value has hyperbolic characteristics [31] corresponding to a saddle point in the RIK solution space, as illustrated by the × in the center of Fig. 4. Joint paths $\mathbf{q}_a(t)$ and $\mathbf{q}_b(t)$ are on different sides of the saddle. Joint path $\mathbf{q}_a(t)$ cannot be continuously deformed along the self-motion manifolds into $\mathbf{q}_b(t)$. As such, $\mathbf{q}_a(t)$ and $\mathbf{q}_b(t)$ are in different homotopy classes.

Since joint paths cannot be deformed across a singularity, the coregular self-motion path is treated as two distinct open paths, $\mathbf{q}_a(\psi)$ and $\mathbf{q}_b(\psi)$, with endpoints adjacent to (but not including) the singularity. Self-motion paths that are adjacent to each other at the bifurcation point are called *bifurcation branches*. The homotopy class of $\mathbf{q}_a(t)$ and $\mathbf{q}_b(t)$ directly depends on which of the two bifurcation branches (dashed or dashed-dotted lines in Fig. 4) is crossed. This bifurcation is captured in the bb-roadmap, where $\mathbf{q}_a(\psi)$ and $\mathbf{q}_b(\psi)$ are associated with bifurcation nodes a and b, respectively.

The bb-roadmap of the RIK solution space of Fig. 4 is given by Fig. 5a, where white nodes correspond to regular self-motion paths, black nodes correspond to coregular self-motion paths, and directed edges (arrows) correspond to sub-task homotopy classes connecting the nodes. Fig. 5a corresponds to a single self-motion manifold bifurcating into two self-motion manifolds, whereas Fig. 5b corresponds to two self-motion manifolds converging into one self-motion manifold (such as the task of Fig. 4 with time reversed). There are two routes[5] that traverse the roadmap in Fig. 5a: 1) [*S, a,* T_a] and 2) [*S, b,* T_b], and two routes that traverse the roadmap in Fig. 5b: 1) [S_a*, a, T*] and 2) [S_b*, b, T*].

Fig. 5. Bifurcation branch roadmap of a task path crossing a single coregular value. a) The roadmap of a task starting with 1 self-motion manifold and terminating with 2 self-motion manifolds. b) The roadmap of a task starting with 2 self-motion manifolds and terminating with 1 self-motion manifold.

As a general task progresses, the self-motion manifolds alternate between splitting and joining at the coregular values. Given the open manifolds of Fig. 3, one might assume that the number of homotopy classes, as the task progresses, doubles only at coregular values which split the self-motion manifolds. This assumption is not valid for closed self-motion manifolds as shown below.

4.2. Tasks crossing two coregular values

For a task path crossing two coregular values, consider the solution space between (and including) the coregular values. The task path passes through a single $W-$ sheet. As such, the self-motion manifolds of the interior task points are closed paths, homotopic to each other, but they are not homotopic to the coregular self-motion paths (bifurcation branches) that are *open* paths. The structure the RIK solution space for the task depends on the number of self-motion manifolds of task configurations inside the $W-$ sheet. The bb-roadmap for $W-$ sheets, with two self-motion manifolds (Case R^{C_2}) is simpler than that for one (Case R^{C_1}) and is presented first.

4.2.1. Case R^{C_2}

For a task path through a $W-$ sheet with 2 self-motion manifolds, the RIK solution space has the structure of Fig. 6a with two disjoint cylinders, each with "pinched" ends, joining only at the pinch points. The pinching effect is illustrated in the bottom cylinder by the three self-motion manifolds (solid closed curves, but dotted

when the curve is hidden behind another surface). As a self-motion manifold approaches a coregular self-motion path, the smooth curve is “pinched” to have a sharp corner at the singularity.

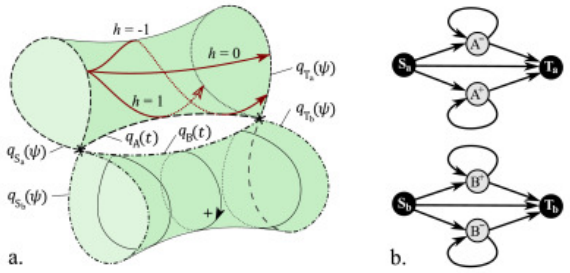

Fig. 6. Structure of the RIK solution space for a task within a $W-$ sheet with two self-motion manifolds. a) The RIK solution surface, two cylinders with “pinched” ends. Dashed and dash-dotted lines represent coregular self-motion paths and singularity connection paths used as roadmap nodes. b) Roadmap where black nodes are coregular self-motion paths of the bifurcation branches and gray nodes are crossings of a singularity connection path. Each route identifies a unique homotopy class.

Consider a joint path starting in coregular self-motion branch $\mathbf{q}_{S_a}(\psi)$. The joint path necessarily terminates in the coregular self-motion path $\mathbf{q}_{T_a}(\psi)$. The joint path may travel along the cylinder “directly” or by “wrapping around” the cylinder clockwise or counter-clockwise, as illustrated by the solid/dotted lines with arrows at the terminal endpoint in Fig. 6a. These paths are homotopically distinct because they cannot be continuously deformed into each other due to the sharp corner at the kinematic singularity.

The homotopy class of a joint path can be identified using the following method:

1. Identify a *singularity connection path*, a path over the RIK solution space that connects the singularities at each end of the task. For some cases, a path may be obtained from the inverse kinematic solution of an equivalent non-redundant mechanism with two twists (columns of the Jacobian matrix) constrained to be linearly dependent.
2. Assign a positive and negative direction for crossing the singularity connection path.
3. Identify all instances at which the joint path crosses the singularity connection path, keeping track of the crossing direction. (This information may be identified using root finding.)
4. Beginning with $h = 0$, for each crossing of the singularity connection path; $h = h + 1$ for a positive crossing, or $h = h - 1$ for a negative crossing. The net-value of h, along with the coregular self-motion branches, identifies the homotopy class of the joint path. The h value of each path along the top cylinder is shown in Fig. 6a.

The RIK solution space of Fig. 6a has two singularity connection paths $\mathbf{q}_a(t)$ and $\mathbf{q}_B(t)$, one on each cylinder. The self-motion path on the lower cylinder with the arrow shows the assigned positive direction for crossing $\mathbf{q}_B(t)$.

The roadmap[6] in Fig. 6b completely captures the homotopy classes for this case. Gray nodes indicate the direction in which a singularity connection path is crossed. The node letter (*A, B*) identifies which path and the superscript sign (+ , -) identifies the crossing direction. Cycles in the roadmap identify joint paths with self-motion cycles. A joint path that “wraps around” the entire cylinder performs a self-motion cycle and is conservatively identified by the joint path crossing the singularity connection path twice (in the same direction). For example, the route containing one cycle: $[S_a, A^+, A^+, T_a]$,, corresponds to a path with $h_A = 2$ that makes at least one full self-motion cycle, but less than two self-motion cycles.

Since it is extremely unlikely that the globally optimal path for any fixed endpoint combination would contain a full self-motion, the bb-algorithm uses a simplified roadmap without gray nodes associated with self-motion cycles. The bb-roadmap of this case is shown in Fig. 8e, where each edge corresponds to a homotopy class without a self-motion cycle. There are 6 different homotopy classes, 3 for each starting node. Each bb-edge number in Fig. 8e has a corresponding node sequence (summarized in Table 1).

Table 1. Homotopy Classes of R^{C_2}.

Edge in Fig. 8e	Node Sequence in Fig. 6b
1	S_a, T_a
2	S_a, A+, T_a
3	S_a, A−, T_a
4	S_b, T_b
5	S_b, B+, T_b
6	S_b, B−, T_b

4.2.2. Case R^{C_1}

For a task path through a W − sheet with a single self-motion manifold, the RIK solution space has the structure of Fig. 7a (one cylinder with "pinched" ends). The pinching effect is illustrated by the three self-motion manifolds (solid/dotted closed curves). As a self-motion manifold approaches a coregular self-motion path, the manifold becomes pinched and eventually two opposite points on the manifold join.

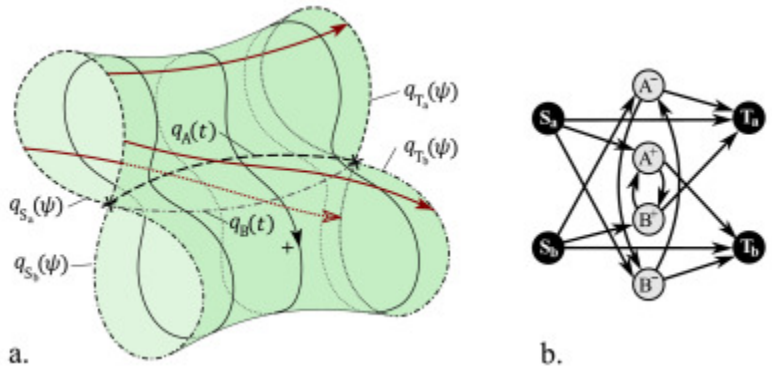

Fig. 7. Structure of the RIK solution space for a task within a W − sheet with one self-motion manifold. a) The RIK solution surface, a cylinder with "pinched" ends. b) Roadmap where black nodes are coregular self-motion paths and gray nodes are crossings of a singularity connection path. Each route identifies a unique homotopy class.

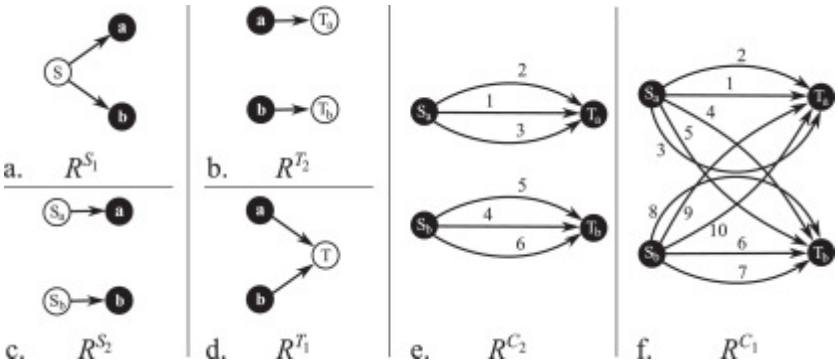

Fig. 8. Bifurcation branch roadmaps of sub-tasks bounded by coregular values or a complete task endpoint. a–d) Sub-task bb-roadmaps with regular self-motion manifolds of a task endpoint. e–f) Sub-task bb-roadmaps with coregular self-motion paths at both endpoints.

Unlike the previous case in which two singularity connection paths existed in separate RIK solution surfaces, this case has two singularity connection paths ($\mathbf{q}_a(t)$ and $\mathbf{q}_B(t)$) on the same RIK solution surface.

Consider a joint path starting in a given coregular self-motion branch, $\mathbf{q}_{S_a}(\psi)$. As in the previous case, a "direct" joint path to $\mathbf{q}_{T_a}(\psi)$ exists ($h = 0$). Unlike the previous case, the joint path here may also travel to the other coregular self-motion branch, $\mathbf{q}_{T_b}(\psi)$ by crossing singularity connection path $\mathbf{q}_a(t)$ or $\mathbf{q}_B(t)$, as illustrated in the example paths in Fig. 7a.

The roadmap in Fig. 7b completely captures the homotopy classes for this case. Removing the self-motion cycles yields the bb-roadmap of this case, shown in Fig. 8f, where each edge corresponds to a unique homotopy class. There are 10 different homotopy classes, 5 for each starting node. Each bb-edge number in Fig. 8f has a corresponding node sequence (summarized in Table 2).

Table 2. Homotopy Classes of R^{C_1}.

Edge in Fig. 8f	Node Sequence in Fig. 7b
1	S_a, T_a
2	S_a, A+, B+, T_a
3	S_a, B−, A−, T_a
4	S_a, A+, T_b
5	S_a, B−, T_b
6	S_b, T_b
7	S_b, A−, B−, T_b
8	S_b, B+, A+, T_b
9	S_b, A−, T_a
10	S_b, B+, T_a

4.3. General tasks

Consider a general task path that crosses multiple coregular values. The task is divided at the coregular values into a series of sub-task paths.

The bb-roadmap of a general task is a concatenation of the sub-task bb-roadmaps shown in Fig. 8. Each sub-task roadmap is labeled R^{Sw}, R^{Tw}, or R^{Cw}, where *w* is the number of distinct self-motion manifolds of the $W-$ sheet containing the sub-task path, the superscripts *S, T*, and *C* correspond to start, terminal, and coregular endpoints, respectively.

The sub-task bb-roadmaps are always combined at the coregular self-motion nodes. For example, the simple bb-roadmap in Fig. 5a is constructed by combining R^{S_1} (Fig. 8a) and R^{S_2} (Fig. 8b). For a general task, the sub-task roadmaps alternate between $w = 1$ and $w = 2$. If the task has free boundary conditions, the first sub-task roadmap is R^{Sw} and the last sub-task roadmap is R^{Tw}. If the task has periodic boundary conditions, for path planning purposes, the start point can be selected to be at a coregular value such that every sub-task roadmap is R^{Cw}.

The number of homotopy classes for a general task path is given by the number of admissible routes through the bb-roadmap. For cyclic tasks, when periodic boundary conditions are required, the start and terminal joint configuration must be in the same coregular self-motion branch. For free boundary conditions, there is no constraint on the start or terminal self-motion node, and since the bb-roadmap is symmetric, the number of homotopy classes is

(6)

$$N_H = 2\prod_{k=2}^{K-1} N_h(R_k),$$

where K is the number of sub-tasks in a complete task sequence, and $N_h(R_k)$ is the number of sub-task homotopy classes per start node in the k^{th} sub-task. For the sub-task roadmaps for closed manifolds considered above, $N_h = 3$ for R^{C_2}, and $N_h = 5$ for R^{C_1}. The number of sub-task homotopy classes is independent of the type of sub-task roadmaps for $k = 1$ and $k = K$. The total number of homotopy classes increases rapidly as the number of coregular value crossings increases.

4.4. Generating node descriptions

Nodes of the bb-roadmap correspond to continuous self-motion paths. These paths must be mathematically defined such that representative samples of joint configurations can be obtained.

Some redundant robots have analytical expressions for identifying self-motion paths of bb-nodes. However, most robot structures do not have an analytical self-motion parameter that is valid over the *entire* workspace of the robot [32]. For this reason a "natural" parametrization corresponding to arc length on the joint configuration manifold is used to define the self-motion path.

The self-motion path $\mathbf{q}(\psi)$ for a closed manifold is numerically generated starting from an initial joint configuration $\mathbf{q} = \mathbf{q}_0$ at $\psi = 0$. A finite joint motion along the null space is identified using:

(7)

$$\Delta\mathbf{q}_N = \mathbf{W}^{-1/2}\hat{\mathbf{n}}\Delta\psi_{\mathrm{d}},$$

where $\mathbf{W}$ is a positive definite weighting matrix that defines the distance metric in the joint tangent space, $\hat{\mathbf{n}}$ is the unit null space vector of the Jacobian matrix of the current joint configuration $\mathbf{q}$ and $\Delta\psi_{\mathrm{d}}$ is the desired arc length step size.

A new joint configuration $\mathbf{q}' \approx \mathbf{q} + \Delta\mathbf{q}_N$ is refined using the Newton–Raphson method to eliminate task error. The arc length distance traveled in that step is estimated using

(8)

$$\Delta\psi = \sqrt{\Delta\mathbf{q}^T\mathbf{W}\Delta\mathbf{q}},$$

where $\Delta\mathbf{q} = \mathbf{q}' - \mathbf{q}$. For small arc length step sizes $\Delta\psi_{\mathrm{d}} \approx \Delta\psi$.

The new joint configuration $\mathbf{q}'$ at $\psi + \Delta\psi$ is saved in a self-motion sequence. The current joint configuration is updated q'→q and the process repeats until the path returns to the initial configuration $\mathbf{q}_0$,, completing the self-motion cycle.

The self-motion path, numerically generated as a sequence of configurations, is converted into a continuous path using a cubic spline fit.

For coregular self-motion paths, the singularity configuration is usually identified by a known geometric condition and can be used as the initial point. However, (7) is not valid at the kinematic singularity. The two directions for feasible finite self-motion must be identified using another method such as that presented in [33] or using an analytical parametrization. Once two nearby configurations are identified by taking a small step in the identified directions, the coregular self-motion paths can be generated using the method described above. Two paths are generated (e.g., $\mathbf{q}_a(\psi)$ and $\mathbf{q}_b(\psi)$ in Fig. 4), each path returns to the singularity.

4.5. Roadmap construction discussion

The bifurcation branch roadmap (bb-roadmap) depends on the characteristics of the manipulator's self-motion manifolds and their bifurcations. The evaluation presented above for different sequences of self-motion manifold bifurcations and their impact on the number of joint path homotopy classes applies to manipulation having either one or two closed self-motion manifolds. An equivalent evaluation would be needed to determine the bb-roadmaps for manipulators with greater numbers of self-motion manifolds.

The only analysis required for constructing the bb-roadmap for a general task is identifying the number of coregular values and the number of self-motion manifolds of task points between them. The number of homotopy classes is immediately identified as the number of routes through the bb-roadmap.

When the conditions for coregular values are known, the time instances of the coregular values along the task path are identified using root finding. The self-motion paths of the bb-nodes at these task instances, along with the task endpoints, are generated using the self-motion path generation method described above. These self-motion paths are used in the remaining steps of the bb-algorithm to search for the globally optimal joint path.

5. Bi-directional instantaneous path planner

This section introduces a new instantaneous path planning procedure used in the bifurcation branch algorithm in two different steps.

The new instantaneous path planner is used first in Step 2: Initial Path Network Generation to connect two sampled joint configurations. The instantaneous planner ensures that the generated path is in the appropriate sub-task homotopy class.

The instantaneous path planner is again used in Step 4: Refined Path Network Generation in alternating sub-tasks to connect the terminal point of one locally optimal path in the prior sub-task to the start point of a locally optimal path in the following sub-task. The instantaneous path planner ensures that piecewise concatenation of both instantaneously generated paths and locally optimal paths yields a complete joint path that is continuous, smooth, and satisfies the task equality constraints.

5.1. Path calculation

The instantaneous path planner bi-directionally integrates (2), in which a non-zero "preferred" joint motion $\dot{\mathbf{q}}_N$ is used and continuously updated to guide the generated joint paths to meet.

Consider the desired endpoint joint configurations $\mathbf{q}_A$ and $\mathbf{q}_B$ at times t_A and t_B, respectively. The bi-directional planner simultaneously generates a forward-growing joint path $\mathbf{q}_a(\tau)$ starting at $\mathbf{q}_a(0) = \mathbf{q}_A$ and a backward-growing a joint path $\mathbf{q}_b(\tau)$ starting at $\mathbf{q}_b(0) = \mathbf{q}_B$ by integrating

(9)

$$\begin{bmatrix} \dot{\mathbf{q}}_a(\tau) \\ \dot{\mathbf{q}}_b(\tau) \end{bmatrix} = \begin{bmatrix} \mathbf{J}^{\dagger}(\mathbf{q}_a)\dot{\mathbf{x}}(\tau + t_A) + s(\tau)\mathbf{P}(\mathbf{q}_a)\dot{\mathbf{q}}_N \\ \mathbf{J}^{\dagger}(\mathbf{q}_b)(-\dot{\mathbf{x}}(t_B - \tau)) - s(\tau)\mathbf{P}(\mathbf{q}_b)\dot{\mathbf{q}}_N \end{bmatrix}$$

over $\tau \in [0, (t_A + t_B)/2]$ where, $\mathbf{P} = (\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J})$,, and $s(\tau)$ is a monotonic scaling function that ranges from 0 to 1 over τ. The preferred joint motion $\dot{\mathbf{q}}_N$ is a direct velocity to go from path $\mathbf{q}_a(\tau)$'s current terminal point $\mathbf{q}_a$ to path $\mathbf{q}_b(\tau)$'s current terminal point $\mathbf{q}_b$ given by

(10)

$$\dot{\mathbf{q}}_N = \frac{\mathbf{q}_b - \mathbf{q}_a}{(t_B - t_A) - 2\tau},$$

where $(t_B - t_A) - 2\tau$ is the normalized time difference between the two converging paths. The scaling function $s(\tau)$ is used to cause the paths to satisfy boundary conditions and smoothly meet. At the start of paths $\mathbf{q}_a(\tau)$ and $\mathbf{q}_b(\tau)$, the instantaneous joint motion minimizes the weighted norm velocity that satisfies free boundary conditions [4]. By the end of the path generation, the joint motion is selected to go directly to the updated terminal point of the other path to connect the paths.

The resulting joint path is

(11)

$$\mathbf{q}_{a\to b}(t) = \begin{cases} \mathbf{q}_a(t - t_A), & \text{for} t_A \le t \le \dfrac{t_A + t_B}{2} \\ \mathbf{q}_b(t_B - t), & \text{for} \dfrac{t_A + t_B}{2} < t \le t_B. \end{cases}$$

If there exists a joint path homotopy between $\mathbf{q}_A$ and $\mathbf{q}_B$ and there is no self-motion bifurcation between them, the bi-directionally generated joint paths will converge (the terminal points of $\mathbf{q}_a(\tau)$ and $\mathbf{q}_b(\tau)$ are identical) to form a continuous path. The joint paths smoothly connect when $s(\tau)$ approaches 1 with a slope of zero. For instance, $s(\tau)$ may be a cubic-spline with slopes clamped at zero.

5.2. Piecewise construction of complete initial paths

A complete initial joint path through the RIK solution space is constructed by piecing together instantaneously generated paths or locally optimal paths (of sub-tasks) associated with the route edges of the configuration roadmap produced in Step 4: Refined Path Network Generation of the bb-algorithm. Since the instantaneously generated paths satisfy fixed boundary conditions and have the instantaneous joint velocity required for free boundary conditions, they *smoothly* connect to locally optimal paths of the sub-task homotopy classes.

The endpoints of the initial complete joint path can be controlled to satisfy fixed or periodic boundary conditions. For fixed boundary conditions, the start and terminal nodes are selected to be the boundary condition configurations. For periodic boundary conditions, the start and end nodes are duplicate instances of the same joint configuration.

5.3. Challenges for closed self-motion manifolds

The bb-roadmap presented in Section 4 for closed self-motion manifolds has multiple edges connecting the same nodes representing multiple sub-task homotopy classes bounded by the same self-motion paths. The joint paths generated with the bi-directional instantaneous path must be associated with the proper sub-task homotopy class. The sub-task homotopy class of the joint path is identified based on the net crossings of the relevant singularity connection path(s). This information can be obtained using root finding when the singularity connection path is generated with the inverse kinematic solution of an equivalent non-redundant mechanism with two twists (columns of the Jacobian matrix) constrained to be linearly dependent.

A single use of the bi-directional path planner between two fixed points easily finds the “direct” path through the RIK solution space between coregular values. However, finding paths that cross a singularity connection path, may require additional control. To control which singularity connection path is crossed, an intermediate

point is selected on the relevant singularity connection path and two paths are generated (start to intermediate and intermediate to terminal) and pieced together.

6. Path deformation procedure

This section describes the procedure used for deforming a sub-optimal joint path into a locally optimal joint path. Although the procedure is restricted to problems with one degree of redundancy, it reliably converges to a locally optimal joint path even if a very high degree of deformation is required. The procedure also ensures that the joint path will remain in the same homotopy class and tracks the task path with a high degree of accuracy. The method is applicable for fixed, free, and periodic boundary conditions.

Given an initial joint path $\mathbf{q}_0(t)$ for the task path $\mathbf{x}(t)$,, a locally optimal path is found using the following steps:

1. Discretize the initial joint path into a set of configurations with corresponding time indicies:

   (12)
   $$\mathbf{q}_0(t) \rightarrow \{\begin{matrix}\mathbf{q}_{0_1} & \mathbf{q}_{0_2} & \dots & \mathbf{q}_{0_k} \\ t_1 & t_2 & \dots & t_k.\end{matrix}\}$$
   This set of joint configurations should include joint configurations associated with the bb-nodes: the joint configurations at task endpoints and the joint configurations on each of the intersecting bifurcation branches. Additional joint configurations evenly spaced between bb-nodes are included to accurately approximate the task path.
2. For each joint configuration $\mathbf{q}_{0_i}$,, ($i \in [1,2,\dots,k]$), generate a self-motion path $\mathbf{q}_i(\psi_i) \mid \mathbf{f}(\mathbf{q}_i(\psi_i)) = \mathbf{x}(t_i), \mathbf{q}_{0_i} \in \mathbf{q}_i(\psi_i)$,, where $\psi_i \in [\psi_{i_{\text{min}}}, \psi_{i_{\text{max}}}]$ is a bounded self-motion parameter.
3. Select the optimization parameters as positions on the bounded self-motion paths $u_i \in [\psi_{i_{\text{min}}}, \psi_{i_{\text{max}}}]$. Every self-motion path ($i \neq 1, k$), is an independent optimization parameter $\psi_i \rightarrow u_i$. If the task endpoint ($i = 1, k$) has a free boundary condition, then $\psi_i \rightarrow u_i$. If the boundary condition is fixed, ψ_i is a fixed value (e.g., $\psi_i = 0$). If the boundary conditions are periodic, then $\psi_1 \rightarrow u_1$ and $\psi_k = u_1$.
4. Use nonlinear programming (NLP), such as an interior-point algorithm [34], to solve:

   (13)
   $$\text{min.} \quad \int_{t=0}^{t=1} \frac{1}{2} \dot{\mathbf{q}}(\mathbf{u}, t)^T \mathbf{W} \dot{\mathbf{q}}(\mathbf{u}, t) dt$$
   $$\text{s.t.} \quad \mathbf{u}_{\text{min}} \leq \mathbf{u} \leq \mathbf{u}_{\text{max}},$$
   where $\mathbf{W}$ is a positive definite weighting matrix that defines the distance metric on the joint configuration manifold and $\mathbf{q}(\mathbf{u}, t)$ is a joint motion path expressed as a cubic-spline with k nodes defined by $\mathbf{u}$ (the set of optimization parameters). For periodic boundary conditions, the start and terminal slopes of the joint path spline are constrained to be the same to enforce continuity in the joint velocities.

The novelty in this modified direct approach is the use of self-motion paths. Using self-motion paths: 1) eliminates the need to include the equality constraints of the task path, 2) reduces the number of dimensions to specify the node locations in the joint space, and 3) allows for large, nonlinear deformation of the nodes for each iteration of the NLP solver. Because self-motion paths of bifurcation branches are used, the NLP solver can

deform the path arbitrarily close to a bifurcation point (kinematic singularity) without crossing it. Also, because the self-motions are parameterized by arc length on the joint configuration manifold, the objective function is not highly sensitive to changes in the optimization parameters. The algorithm is therefore both fast and reliable and it is incapable of "jumping" the path across a bifurcation point into a different homotopy class.

The self-motion paths generated in Step 2 of the path deformation procedure are generated bi-directionally outward from the initial joint configuration $\mathbf{q}_{0_i}$. One step of the self-motion path generation is illustrated at the first node in Fig. 9. Starting at the base-point, there is a "positive" motion direction given by $\hat{\mathbf{n}}$ and a "negative" motion direction given by $-\hat{\mathbf{n}}$. The self-motion nodes are generated in both directions until bounds are reached at $\psi_{i_{\max}}$ and $\psi_{i_{\min}}$. The bounds may be selected to be some arbitrary maximum arc length along the self-motion manifold from the initial point (e.g., half the arc length of the nearest bifurcation branch).

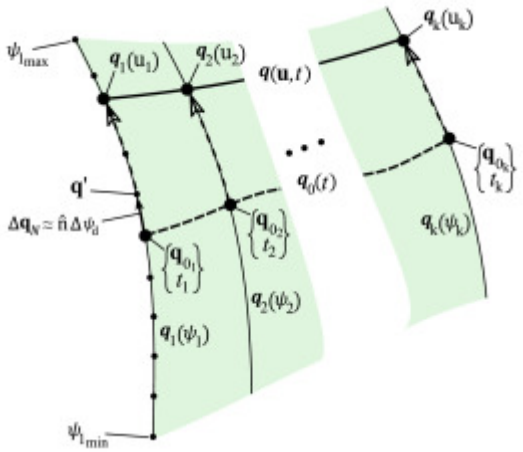

Fig. 9. Joint path deformation on an RIK solution surface. The initial joint path (thick dashed line) is discretized (large dots). The nodes are translated along the self-motion paths (thin solid lines), and the deformed path (thick solid line) is approximated with a cubic spline through the large dots.

If the NLP solver converges to a joint path that does not touch the self-motion bounds, the resulting path is locally optimal. If the NLP solver converges to a path that touches the self-motion bounds, the resulting path is used as an initial path and steps 1–4 are repeated until a locally optimal path is reached.

The equality constraints of (4) associated with the task path are accounted for by the self-motion paths. These constraints are satisfied at the discrete points, but not between them. For instance, a spline containing $\mathbf{q}_1(u_1)$, $\mathbf{q}_2(u_2)$, and $\mathbf{q}_k(u_k)$ in Fig. 9 will detach slightly from the actual solution surface between the self-motion paths. The task error of a joint path is evaluated using

(14)

$$x_{\text{err}} = \int_{t=0}^{t=1} (\mathbf{x}(t) - \mathbf{f}(\mathbf{q}(\mathbf{u},t))^T (\mathbf{x}(t) - \mathbf{f}(\mathbf{q}(\mathbf{u},t))\, dt,$$

where the numerical integration uses a mesh at least 20 times more dense than the joint path discretization. If the task error exceeds a specified threshold value, more nodes are added to u, those self-motion paths are evaluated, and the interior-point algorithm is restarted. The selection of the task error tolerance should ensure that the global cost value of the path is insensitive to tightening the task error tolerance. A good task error tolerance can be selected from a sensitivity analysis by deforming the best joint path from Step 4: Refined Path Network Generation using a different number of discretization points and comparing the task error values and global cost values.

The run-time of the bb-algorithm is closely coupled to the task error tolerance. The use of iterative refinement increases the robustness of the bb-algorithm (confidence that the output is the globally optimal joint path), but adds variability in the computation time.

7. Case study

This section demonstrates the bifurcation branch algorithm (bb-algorithm) for finding the globally optimal joint path for a 3R manipulator executing particle planar task paths. First, the self-motion and singularity characteristics are described to show that the sub-task roadmaps presented in Section 4 are valid for any task path in the manipulator's workspace. Next, the RIK solution spaces of three simple tasks are presented. These tasks have solution space structures like those in Fig. 4, Fig. 6, and 7. Lastly, a complex task path is presented for which there is a very high number of homotopy classes and the globally optimal path found using the bb-algorithm is not found using traditional methods.

7.1. Manipulator

Consider the 3R planar manipulator depicted in Fig. 10 performing particle planar tasks. The normalized link lengths (dimensionless proportions of its total length L) are $l_1 = \frac{6}{15}$, $l_2 = \frac{5}{15}$, and $l_3 = \frac{4}{15}$, the same proportions used in [5]. The manipulator configuration is described by $\mathbf{q} = [q_1, q_2, q_3]^T$ as illustrated in Fig. 10. The manipulator is ceiling-mounted such that there is no possibility for self-collision.

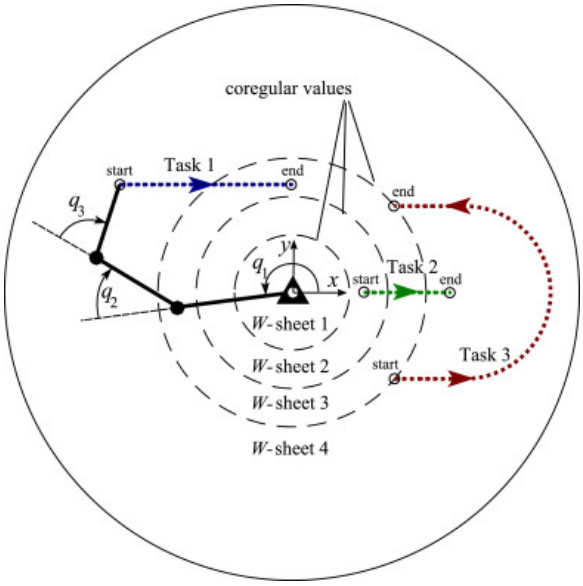

Fig. 10. A 3R manipulator configuration in its workspace. The workspace is divided into $W-$ sheets, bounded by coregular values. Task points in $W-$ sheets, 2 and 4 have a single self-motion manifold, while task points in $W-$ sheets, 1 and 3 have two disjoint self-motion manifolds.

Each joint motion $\dot{q}_i$ is controlled by an actuator motion $\dot{\phi}_i$ related by a transmission ratio: $\dot{\phi}_i = \rho_i \dot{q}_i$. Consider identical actuators for each joint, but with different transmission ratios, $\rho_1 = l_1 + l_2 + l_3, \rho_2 = l_2 + l_3, \rho_3 = l_3$, each based on the total link length beyond the joint. The selected optimization criterion is minimizing the actuator velocity norm, i.e., minimizing: $\| \dot{\boldsymbol{\phi}} \|^2 = \dot{\mathbf{q}}^T \mathbf{W} \dot{\mathbf{q}}$, where

(15)

$$\mathbf{W} = \begin{bmatrix} \rho_1^2 & 0 & 0 \\ 0 & \rho_2^2 & 0 \\ 0 & 0 & \rho_3^2 \end{bmatrix}.$$

The 3R manipulator configuration is singular if all three links are aligned. The kinematic singularities internal to the workspace have joint configurations at $\mathbf{q} = [q_1, \pi, 0]^T$, $\mathbf{q} = [q_1, \pi, \pi]^T$, and $\mathbf{q} = [q_1, 0, \pi]^T$, with any value for q_1. These singular cases yield end-effector configurations (coregular values) at distances $d_1 = |l_1 - l_2 - l_3|$, $d_2 = |l_1 - l_2 + l_3|$, and $d_3 = |l_1 + l_2 - l_3|$ from the base. The set of coregular values bounding the $W -$ sheets, are identified by the dashed rings in Fig. 10.

$W -$ sheet 1 has two closed self-motion manifolds corresponding to the kinematics of a double-crank 4-bar mechanism with "elbow-up" and "elbow-down" poses. *W*-sheet 3 has two closed self-motion manifolds corresponding to the kinematics a crank-rocker 4-bar mechanism with "elbow-up" and "elbow-down" poses. $W -$ sheet 2 and $W -$ sheet 4 each have one closed self-motion manifold that corresponds to the kinematics of a 4-bar mechanism that cannot be fully driven by a single input link.

Each $W -$ sheet is bounded by a $W -$ sheet with a different number of self-motion manifolds. As such, for an arbitrary task path in the workspace, all regular task configurations have either one or two *closed* self-motion manifolds, and the number changes whenever a coregular value is crossed. The sub-task bifurcation branch roadmaps for closed self-motion manifolds presented in Section 4 are used in this case study. The exact locations of coregular values on a specified task path $\mathbf{x}(t)$, are identified by finding the roots of $\sqrt{\mathbf{x}(t)^T\mathbf{x}(t)} - d_1$, $\sqrt{\mathbf{x}(t)^T\mathbf{x}(t)} - d_2$, and $\sqrt{\mathbf{x}(t)^T\mathbf{x}(t)} - d_3$.

7.2. Relatively simple tasks

Tasks 1, 2, and 3, shown in Fig. 10, were selected to show important features of the bb-algorithm and to illustrate real RIK solution spaces having the structural characteristics described in Section 4. Task 1 has a bifurcation singularity with a saddle point easily visible on its RIK solution space (the same structure as Figs. 4). Task 2 has the same RIK structure as Fig. 6 and has multiple locally optimal joint paths (and multiple within the same homotopy class), all of which are found by the bb-algorithm. Task 3 has the same RIK structure as Fig. 7 and is used to show multiple sub-optimal joint paths in different homotopy classes, each generated using the bi-directional instantaneous path planner.

7.2.1. Task 1

Task 1 in Fig. 10 crosses a single coregular value. It starts at a task point with one self-motion manifold $\mathbf{q}_S(\psi)$, and ends at a task point with two self-motion manifolds $\mathbf{q}_{T_a}(\psi)$, and $\mathbf{q}_{T_b}(\psi)$. The RIK solution space is shown as a sequence of self-motion manifolds in Fig. 11, where Fig. 11a is a top view, and Fig. 11b is a trimetric view. The top view looks like the structure in Fig. 4a, in which the coregular self-motion (bold curve) is self-intersecting. The self-motion paths at time values after the coregular value, are pairs of "elbow-up" and "elbow-down" manifolds. These self-motion paths do not look like closed curves in Fig. 11b, but they are closed because when the manipulator performs a self-motion cycle, it returns to the same configuration after traveling 2π over q_3. The self-motion paths at time values before the coregular value are closed curves in Fig. 11b and are only shown for a single instance, though there are multiple instances separated by 2π over each joint coordinate q_i. For task paths crossing multiple coregular values, it is important to account for equivalent self-motion manifolds separated by 2π over q_i.

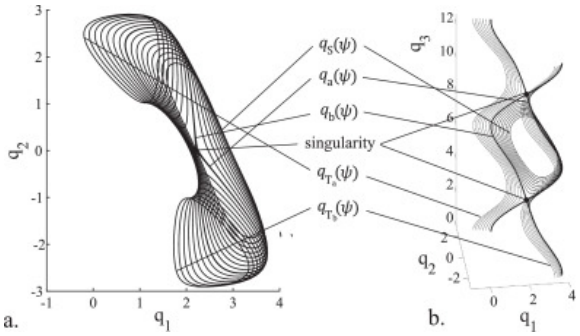

Fig. 11. The RIK solution space of Task 1 shown as a sequence of self-motion paths. a) topview. b) trimetric view. Thin curves are regular self-motion paths. Thick curves are coregular self-motion paths.

7.2.2. Task 2

Task 2 in Fig. 10 crosses two coregular values and there are two disjoint self-motion manifolds for task points between the coregular values. The portion of the task between the coregular values has an RIK solution space with the same structure as that shown in Fig. 6 (if the "cylinders" in Fig. 6 were cut at the singularity connection paths and unrolled). Task 2, however, extends beyond the coregular values to better show different homotopy classes.

The RIK solution space of Task 2 is shown in Fig. 12. A joint path starting in the single self-motion manifold $\mathbf{q}_S(\psi)$, can either pass through coregular self-motion $\mathbf{q}_{1_a}(\psi)$ or $\mathbf{q}_{1_b}(\psi)$. Joint path passing through $\mathbf{q}_{1_a}(\psi)$, must also pass through $\mathbf{q}_{2_a}(\psi)$ traveling through the "elbow-up" RIK solution space. The dashed and dashed-dotted lines in Fig. 12 are the singularity connection paths obtained from reduced inverse kinematics constraining $q_3 = \pi$ to make twists 2 and 3 linearly dependent. There are two inverse kinematic solutions; one solution traces the singularity connection path on the "elbow-up" solution space and the other solution traces the singularity connection path on the "elbow-down" solution space. Both singularity connection paths are shown in two instances, separated by a difference of 2π in q_3.

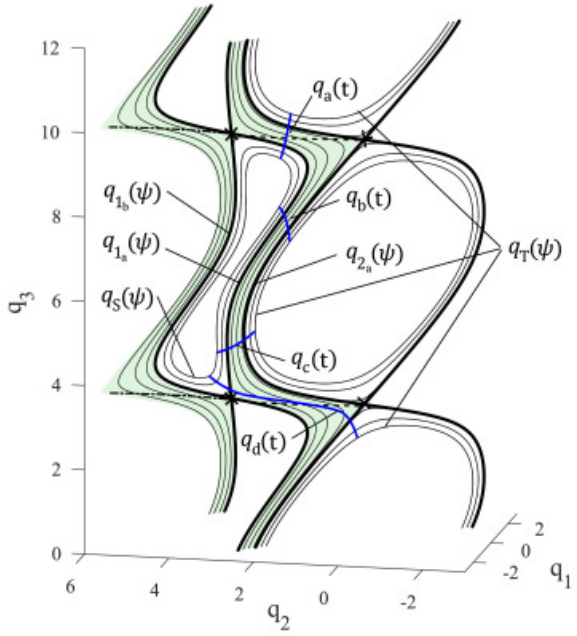

Fig. 12. The RIK solution space of Task 2. The shaded regions are disconnected solution spaces for the sub-task between the coregular values. Thin curves are regular self-motion paths. Thick curves are coregular self-motion

paths or locally optimal joint paths depending on the label. Dashed and dashed-dotted lines are singularity connection paths.

The bb-algorithm found 8 locally optimal joint paths in 6 homotopy classes. Four locally optimal paths ($\mathbf{q}_a(t)$, $\mathbf{q}_b(t)$, $\mathbf{q}_c(t)$, $\mathbf{q}_d(t)$) with free boundary conditions are shown on the "elbow-up" RIK solution surface in Fig. 12. Two of these paths, $\mathbf{q}_b(t)$, and $\mathbf{q}_c(t)$, do not cross the singularity connection path and are in the same homotopy class. It is easily seen that each task point in $\mathbf{q}_b(t)$ can be deformed along the self-motions into $\mathbf{q}_c(t)$. Paths $\mathbf{q}_a(t)$ and $\mathbf{q}_d(t)$ cross the singularity connection path in different directions; $\mathbf{q}_a(t)$ crosses the singularity connection path with an increasing q_3 value, whereas $\mathbf{q}_d(t)$ crosses the singularity connection path with a decreasing q_3 value. Although $\mathbf{q}_a(t)$, $\mathbf{q}_b(t)$, and $\mathbf{q}_d(t)$ terminate in the same self-motion manifold $\mathbf{q}_T(\psi)$, they terminate in manifolds separated by 2π in q_3. These paths cannot be continuously deformed into each other.

7.2.3. Task 3

Task 3 in Fig. 10 has task endpoints at coregular values, all other task configurations are inside *W*-sheet 4 and have a single self-motion manifold. The RIK solution space for this task is illustrated in Fig. 13 and has the same general structure as that of Fig. 7. Smooth joint paths $\mathbf{q}_a(t)$, $\mathbf{q}_b(t)$, and $\mathbf{q}_c(t)$ are generated using the bi-directional path planner, each starting from a joint configuration at the midpoint of the coregular self-motion path. Each of these paths has the same starting configuration and the same initial joint motion, but each tangentially diverges into a different homotopy class.

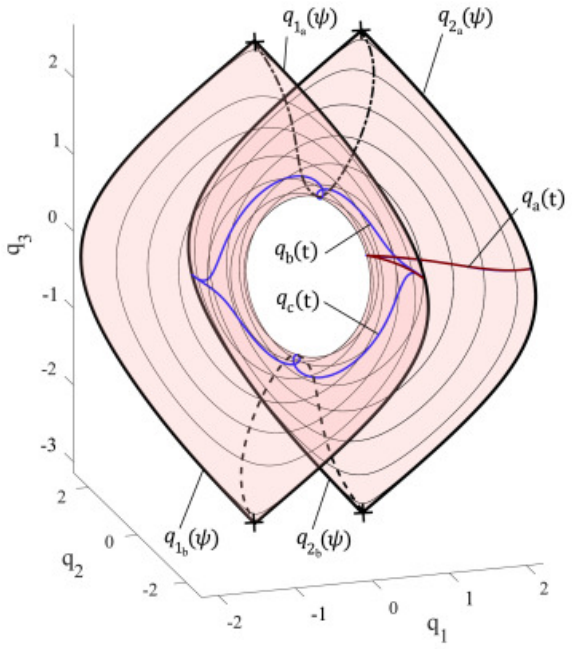

Fig. 13. The RIK solution space of Task 3. Thin curves are regular self-motion paths. Thick curves are coregular self-motion paths or joint paths. Dashed and dashed-dotted lines are singularity connection paths.

Path $\mathbf{q}_a(t)$ is a "direct path" connecting $\mathbf{q}_{1_a}(\psi)$ to $\mathbf{q}_{2_a}(\psi)$ it does not cross a singularity connection path. The dashed and dashed-dotted lines in Fig. 13 are the two singularity connection paths corresponding to reduced inverse kinematic solutions, where link 1 is oriented to point to the end-effector location (twists 1 and 3 are linearly dependent). There are two inverse kinematic solution cases corresponding to "wrist-up" and "wrist-down" cases of the two distal links. Paths $\mathbf{q}_b(t)$ and $\mathbf{q}_c(t)$ connect $\mathbf{q}_{1_a}(\psi)$ to $\mathbf{q}_{2_b}(\psi)$, but are in separate

homotopy classes because they travel different directions around the solution surface, crossing different singularity connection paths.

Because of the symmetry of the solution space, attempting to bi-directionally generate a path between the midpoints of $\mathbf{q}_{1_a}(\psi)$ and $\mathbf{q}_{2_b}(\psi)$ yields bi-directionally generated paths that fail to meet, terminating at opposite ends of the same self-motion manifold. An intermediate point on the singularity connection path is used with the bi-directional path planner to control which way the joint path goes around the solution space.

7.3. General task

Here, the best locally optimal joint paths with free boundary conditions are found using the bb-algorithm and compared to the best path found using a traditional method. Indirect methods (e.g., solving a boundary value problem with shooting methods) fail because the numerical integration is not stable for the complex task path. Instead, a multi-start method is used to obtain multiple sub-optimal joint paths. The multi-start method generates sub-optimal joint paths by integrating (2) with $\dot{\mathbf{q}}_N = \mathbf{0}$ from multiple admissible joint configurations equally-spaced along the self-motion manifold at the task start point. These sub-optimal paths are then deformed into locally optimal paths using the path deformation method described in Section 6.

Consider the complex task path shown in Fig. 14 (defined by a cubic-spline) that crosses 12 coregular values. The bb-roadmap is constructed by linking 13 sub-task roadmaps (of Fig. 8) in the following order:

$$R_1^{S_1}, R_2^{C_2}, R_3^{C_1}, R_4^{C_2}, R_5^{C_1}, R_6^{C_2}, R_7^{C_1}, R_8^{C_2}, R_9^{C_1}, R_{10}^{C_2}, R_{11}^{C_1}, R_{12}^{C_2}, R_{13}^{T_1}$$

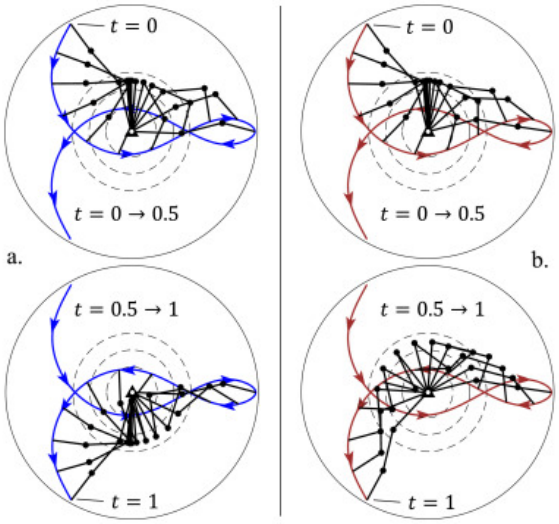

Fig. 14. Stroboscopic image of the manipulator performing the best joint paths found using: a) bb-algorithm, and b) multi-start method (11th best path found by the bb-algorithm). The top images show manipulation of the first half of the task, the bottom images show manipulation of the second half of the task.

The total number of homotopy classes (without self-motion cycles in a single *W*-sheet), calculated using (6), is 4,556,250. The number of promising homotopy classes is reduced to 5626 using the upper/lower bound method described in Section 3. Each initial joint path in the promising homotopy classes was deformed into a locally optimal path. The best of these, identified as the globally optimal path, has a cost value of 40.0.

Fig. 14a shows snapshots of the manipulator tracking the best joint path found using the bb-algorithm. To avoid image clutter, the top image only shows the first half of the task and the bottom image only shows the second half of the task.

A multi-start method was used to generate 100 velocity-based (instantaneous) sub-optimal paths starting from equally spaced configurations on the starting self-motion manifold. Each path was deformed into a locally optimal joint path using the path deformation procedure described in Section 6 (using $k > 50$ nodes to define the cubic spline of the joint path and using a task error threshold of 10−9). With these inputs/tolerances, the path deformation procedure took about 250 times more computation time than the instantaneous path planner. The set of 100 sub-optimal paths converged to a set of 7 unique locally optimal paths, each in a different homotopy class. The multi-start method was repeated using 500 starting points and repeated again using 1000 starting points. Only the same 7 unique locally optimal paths were obtained with 500 and 1000 starting points. The costs of the initial sub-optimal paths ranged from 57.3 to 140.7. The cost values of the deformed (locally optimal) paths ranged from 44.7 to 67.6.

The best path of these paths is shown in Fig. 14b. It is the same as the 11th best path found by the bb-algorithm. The bb-algorithm found 10 unique locally optimal paths that have lower cost values than the best path obtained from a rigorous multi-start method. The bb-algorithm took about 6 times longer than the (1,000 point) multi-start method and found a joint path 10 percent better. Even with a very large number of seeds (starting configurations), the multi-start method failed to identify the globally optimal path.

The cost rates[7] of the best paths found using the two methods are shown in Fig. 15. The best path (found by the bb-algorithm) requires that the manipulator switch from an "elbow-up" pose to an "elbow-down" pose over sub-task 7 ($R_7^{C_1}$) in W − sheet 4. However, crossing from $\mathbf{q}_{6_a}(\psi)$ to $\mathbf{q}_{7_b}(\psi)$ requires more joint motion as indicated by the higher cost rates in the middle of the tasks relative to the 11th best path, which remains in the "elbow-up" configuration. Although, the multi-start optimal path has lower cost rates during the middle of the task ($0.22 < t < 0.66$), it has higher cost rates at time greater than 0.66. This is reflected in Fig. 14, which shows greater joint motion for the multi-start optimal path during the second half of the task.

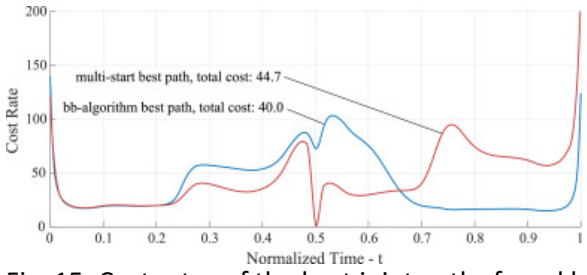

Fig. 15. Cost rates of the best joint paths found by the bifurcation branch algorithm and the multi-start method (11th best path found by the bifurcation branch algorithm).

Paths generated using instantaneous optimization cannot find the homotopy class of the globally optimal path because they are unable to make local sacrifices to reap better returns later in the task.

8. Conclusion

This paper presents an algorithm more capable than traditional methods for finding the globally optimal path of a redundant manipulator performing a task with a defined path. The globally optimal joint path is found using a custom "direct" method for solving optimal control problems. To find the globally optimal path, the direct solver must be initialized with a sub-optimal joint path in the same homotopy class as the globally optimal path. The homotopy class of the globally optimal joint path, however, can be extremely difficult to find without high-level knowledge of the solution space. The bifurcation branch algorithm identifies all homotopy classes, generates many sub-optimal joint paths in each homotopy class, systematically identifies a set of promising homotopy classes that may contain the best path, and deforms the sub-optimal paths in these promising homotopy classes into locally optimal paths. The best locally optimal path found is very likely the globally optimal joint path. A

comparison of the algorithm computational efficiency was not discussed in detail because no other existing algorithm addresses this problem with the same level of completeness.

The bb-roadmap structures of more complex RIK structures with greater number of self-motion manifolds can be identified provided the impact of self-motion bifurcation on the joint path homotopy classes is properly characterized. The concepts introduced in this paper can be extended to higher degrees of redundancy, but new tools for constructing the bb-roadmap and for describing the self-motion manifolds would need to be developed.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work is supported by the National Science Foundation under Grant IIS-1427329.

References

[1] D.E. Whitney. **Resolved motion rate control of manipulators and human prostheses.** IEEE Trans. Man Mach. Syst., 10 (2) (1969), pp. 47-53, 10.1109/TMMS.1969.299896

[2] Y. Nakamura, H. Hanafusa. **Optimal redundancy control of robot manipulators.** Int. J. Robot. Res., 6 (1) (1987), pp. 32-42, 10.1177/027836498700600103

[3] K. Suh, J. Hollerbach. **Local versus global torque optimization of redundant manipulators.** Proceedings of the International Conference on Robotics and Automation, 4, IEEE (1987), pp. 619-624, 10.1109/ROBOT.1987.1087955

[4] K. Kazerounian, Z. Wang. **Global versus local optimization in redundancy resolution of robotic manipulators.** Int. J. Robot. Res., 7 (5) (1988), pp. 3-12, 10.1177/027836498800700501

[5] D.P. Martin, J. Baillieul, J.M. Hollerbach. **Resolution of kinematic redundancy using optimization techniques.** IEEE Trans. Robot. Autom., 5 (4) (1989), pp. 529-533, 10.1109/70.88067

[6] M. Galicki. **The planning of robotic optimal motions in the presence of obstacles.** Int. J. Robot. Res., 17 (3) (1998), pp. 248-259, 10.1177/027836499801700303

[7] B.J. Martin, J.E. Bobrow. **Minimum-effort motions for open-chain manipulators with task-dependent end-effector constraints.** Int. J. Robot. Res., 18 (2) (1999), pp. 213-224, 10.1177/027836499901800206

[8] J. Burdick. **On the inverse kinematics of redundant manipulators: characterization of the self-motion manifolds.** Proceedings of the International Conference on Robotics and Automation, IEEE (1989), pp. 264-270, 10.1109/ROBOT.1989.99999

[9] A. Guigue, M. Ahmadi, M. Hayes, R. Langlois, F. Tang. **A dynamic programming approach to redundancy resolution with multiple criteria.** Proceedings of the International Conference on Robotics and Automation, IEEE (2007), pp. 1375-1380, 10.1109/ROBOT.2007.363176

[10] L.F. Shampine, J. Kierzenka, M.W. Reichelt. **Solving boundary value problems for ordinary differential equations in matlab with BVP4C.** Tutor. Notes, 2000 (2000), pp. 1-27

[11] J.T. Betts. **Practical Methods for Optimal Control and Estimation Using Nonlinear Programming.** 19, SIAM (2010)

[12] E. Schmitzberger, J.L. Bouchet, M. Dufaut, D. Wolf, R. Husson. **Capture of homotopy classes with probabilistic road map.** Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, 3 (2002), pp. 2317-2322, 10.1109/IRDS.2002.1041613

[13] B. Banerjee, B. Chandrasekaran. **A framework of voronoi diagram for planning multiple paths in free space.** J. Exp. Theor. Artif. Intell., 25 (4) (2013), pp. 457-475, 10.1080/0952813X.2012.741625

[14] R. Lavrenov, E. Magid. **Towards heterogeneous robot team path planning: acquisition of multiple routes with a modified spline-based algorithm.** Proceedings of the MATEC Web of Conferences, 113, EDP Sciences (2017), p. 02015, 10.1051/matecconf/201711302015

[15] S. Bhattacharya, M. Likhachev, V. Kumar. **Topological constraints in search-based robot path planning.** Auton. Robot., 33 (3) (2012), pp. 273-290, 10.1007/s10514-012-9304-1

[16] E. Hernandez, M. Carreras, P. Ridao. **A comparison of homotopic path planning algorithms for robotic applications.** Robot. Auton. Syst., 64 (2015), pp. 44-58, 10.1016/j.robot.2014.10.021

[17] J. Park, S. Karumanchi, K. Iagnemma. **Homotopy-based divide-and-conquer strategy for optimal trajectory planning via mixed-integer programming.** IEEE Trans. Robot., 31 (5) (2015), pp. 1101-1115, 10.1109/TRO.2015.2459373

[18] S. Bhattacharya, R. Ghrist. **Path homotopy invariants and their application to optimal trajectory planning.** Ann. Math. Artif. Intell., 84 (3-4) (2018), pp. 139-160, 10.1007/s10472-018-9596-8

[19] X. Wang, S. Bhattacharya. **A topological approach to workspace and motion planning for a cable-controlled robot in cluttered environments.** IEEE Robot. Autom. Lett., 3 (3) (2018), pp. 2600-2607, 10.1109/LRA.2018.2817684

[20] D. Kularatne, S. Bhattacharya, M.A. Hsieh. **Going with the flow: a graph based approach to optimal path planning in general flows.** Auton. Robot., 42 (7) (2018), pp. 1369-1387, 10.1007/s10514-018-9741-6

[21] L.E. Kavraki, P. Svestka, J.C. Latombe, M.H. Overmars. **Probabilistic roadmaps for path planning in high-dimensional configuration spaces.** IEEE Trans. Robot. Autom., 12 (4) (1996), pp. 566-580, 10.1109/70.508439

[22] M. Stilman. **Task constrained motion planning in robot joint space.** Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (2007), pp. 3074-3081, 10.1109/IROS.2007.4399305

[23] D. Berenson, S.S. Srinivasa, D. Ferguson, J.J. Kuffner. **Manipulation planning on constraint manifolds.** Proceedings of the International Conference on Robotics and Automation (2009), pp. 625-632, 10.1109/ROBOT.2009.5152399

[24] M. Stilman. **Global manipulation planning in robot joint space with task constraints.** IEEE Trans. Robot., 26 (3) (2010), pp. 576-584, 10.1109/TRO.2010.2044949

[25] L. Jaillet, J.M. Porta. **Path planning under kinematic constraints by rapidly exploring manifolds.** IEEE Trans. Robot., 29 (1) (2013), pp. 105-117, 10.1109/TRO.2012.2222272

[26] B. Kim, T.T. Um, C. Suh, F.C. Park. **Tangent bundle RRT: a randomized algorithm for constrained motion planning.** Robotica, 34 (1) (2016), pp. 202-225, 10.1017/S0263574714001234

[27] F. Flacco, A. De Luca. **Discrete-time redundancy resolution at the velocity level with acceleration/torque optimization properties.** Robot. Auton. Syst., 70 (2015), pp. 191-201, 10.1016/j.robot.2015.02.008

[28] A. Liégois. **Automatic supervisory control of the configuration and behavior of multibody mechanisms.** IEEE Trans. Syst. Man Cybern., 7 (12) (1977), pp. 868-871, 10.1109/TSMC.1977.4309644

[29] S. Bhattacharya, M. Pivtoraiko. **A classification of configuration spaces of planar robot arms for a continuous inverse kinematics problem.** Acta Appl. Math., 139 (1) (2015), pp. 133-166, 10.1007/s10440-014-9973-1

[30] M. Shimizu, H. Kakuya, W.-K. Yoon, K. Kitagaki, K. Kosuge. **Analytical inverse kinematic computation for 7-DOF redundant manipulators with joint limits and its application to redundancy resolution.** IEEE Trans. Robot., 24 (5) (2008), pp. 1131-1142, 10.1109/TRO.2008.2003266

[31] N.S. Bedrossian. **Classification of singular configurations for redundant manipulators.** Proceedings of the 1990 IEEE International Conference on Robotics and Automation, 1990, IEEE (1990), pp. 818-823, 10.1109/ROBOT.1990.126089

[32] D.R. Baker, C.W. Wampler. **On the inverse kinematics of redundant manipulators.** Int. J. Robot. Res., 7 (2) (1988), pp. 3-21, 10.1177/027836498800700201

[33] J. Kieffer. **Differential analysis of bifurcations and isolated singularities for robots and mechanisms.** IEEE Trans. Robot. Autom., 10 (1) (1994), pp. 1-10, 10.1109/70.285580

[34] R.H. Byrd, J.C. Gilbert, J. Nocedal. **A trust region method based on interior point techniques for nonlinear programming.** Math. Program., 89 (1) (2000), pp. 149-185, 10.1007/PL00011391