A Learning-Based Method for Computing Self-Motion Manifolds of Redundant Robots for Real-Time Fault-Tolerant Motion Planning

Charles L. Clark, *Student Member, IEEE*, and Biyun Xie, *Senior Member, IEEE*

***Abstract*—The focus of this research is to develop a learning-based method that computes self-motion manifolds (SMMs) efficiently and accurately to enable real-time global fault-tolerant motion planning. The proposed method first develops a learnable, closed-form representation of SMMs based on Fourier series. A cellular automaton is then applied to cluster workspace locations having the same number of SMMs and group SMMs with similar shape by homotopy classes, such that the SMMs of each homotopy class can be accurately learned by a neural network. To approximate the SMMs of an arbitrary workspace location, a neural network is first trained to predict the set of homotopy classes belonging to this workspace location. For each set of homotopy classes, another neural network is trained to approximate the Fourier series coefficients of the SMMs, and the joint configurations along the SMMs can be retrieved using the inverse Fourier transform. The proposed method is validated on planar 3R positioning, spatial 4R positioning, and spatial 7R positioning and orienting robots, using 10 000 randomly sampled workspace locations each. The results show that the proposed method can approximate SMMs with high accuracy and is much faster than the traditionally used nullspace projection method, a sampling-based method, and a grid-based method. The performance of the proposed method in real-time fault-tolerant motion planning applications is also demonstrated using the simulation of the spatial 7R robot and physical experiments on a planar 3R robot. Due to the computational efficiency of the proposed method, both robots are able to quickly plan trajectories which maximize the likelihood of task completion after the failure of one arbitrary joint.**

***Index Terms*—Fault tolerance, kinematics, motion and path planning, redundant robots.**

I. Introduction

ROBOTS are an ideal choice to replace human workers in dangerous and remote tasks, such as nuclear waste remediation [1], space exploration [2], and disaster rescue [3]. However, these environments have harsh conditions, such as extreme temperatures, high levels of radiation, and highly unstable structures, which can cause joint failures [4]. Furthermore, it is impossible for humans to safely access these environments to repair the robots after failures. Thus, fault tolerance which is the ability for a robot to complete the remainder of the assigned task after a failure of one arbitrary joint, is critical for ensuring the reliability and robustness of robotic systems operating in these environments. The most frequently occurring joint failures are locked joint failures, which result in a robot's joint becoming immobilized while the other joints can continue operating normally within their joint limits. Other common failure modes, such as free-swinging joint failures can be transformed into locked joint failures by failure recovery mechanisms that employ fail safe brakes [5]. This research work mainly focuses on the locked joint failure mode. Applying kinematically redundant robots is one potential way to realize fault tolerance because they have more degrees of freedom (DOFs) than the minimum DOFs required to complete the assigned tasks. However, kinematic redundancy alone cannot guarantee fault tolerance, thus intelligent motion planning algorithms that optimize performance before and after the failure of one arbitrary joint need to be developed [6].

Fault-tolerant motion planning can be considered in two different phases. The first phase is prefailure motion planning, which maximizes the task completion probability in anticipation of all potential joint failures [7], [8], [9], [10], [11]. The second phase is postfailure motion planning, i.e., failure recovery, which attempts to use a robot's remaining healthy joints to complete the assigned task [12], [13], [14], [15]. In addition, prefailure fault-tolerant motion planning can be classified in two categories. The first category is local fault-tolerant motion planning, which only considers the local fault-tolerant performance of the current configuration, such as the postfailure dexterity [16]. The other category is global fault-tolerant motion planning, which considers the global fault-tolerant performance of the entire trajectory, such as the reachability of the desired task locations [8]. Global fault-tolerant motion planning is typically much more complicated than local fault-tolerant motion planning, and it can be even more difficult when the task locations are not known a priori. This requires real-time global motion planning.

For redundant robots, there are infinitely many configurations associated with a desired task location due to the extra DOFs. All of these possible configurations can be grouped into several distinct manifolds called the self-motion manifolds (SMMs). The SMMs that can be smoothly deformed between one another belong to the same homotopy class. Examples of the SMMs

Received 17 June 2024; revised 15 January 2025; accepted 25 March 2025. Date of publication 9 April 2025; date of current version 29 April 2025. This work was supported in part by the NSF Foundation under Grant #2205292 and in part by the NASA and the NASA Kentucky EPSCoR Program under NASA award number 80NSSC22M0034. This article was recommended for publication by Associate Editor L. Righetti and Editor T. Bretl upon evaluation of the reviewers' comments. *(Corresponding author: Biyun Xie.)*

The authors are with the Electrical and Computer Engineering Department, University of Kentucky, Lexington, KY 40506 USA (e-mail: landon.clark@uky.edu; biyun.xie@uky.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TRO.2025.3559404, provided by the authors.

Digital Object Identifier 10.1109/TRO.2025.3559404

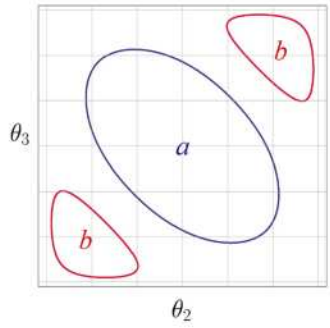

Fig. 1. Example SMMs of a planar 3R robot projected onto the θ_2-θ_3 plane. The purple curve represents the SMM of task point a, while the two red curves represent the two branches of the SMMs belonging to task point b.

belonging to two different workspace locations of an equal link length planar 3R robot are shown in Fig. 1. The purple curve represents the SMM associated with the workspace location a, while the two red curves both represent SMMs associated with the workspace location b. Because none of these three SMMs can be smoothly deformed between each other, there are three homotopy classes in this example. As SMMs represent all of the joint configurations associated with a target location, most prefailure motion planning algorithms rely on SMMs to check the reachability of the task location. Thus, the accurate and efficient computation of SMMs is very important.

The goal of this work is to develop an efficient and accurate learning-based method to approximate the SMMs of robots with one degree of redundancy, and thus the approximated SMMs can be used for real-time global motion planning, such as fault-tolerant motion planning. To accomplish this goal, a learnable, closed-form representation of SMMs is developed by using Fourier series. Because the SMMs of different workspace locations can vary greatly in both number and shape, a cellular automaton is used to cluster workspace locations having the same number of SMMs and group SMMs with similar shapes by homotopy classes. Once the SMMs have been grouped by homotopy classes, a neural network is trained to predict the homotopy classes belonging to an arbitrary workspace location. For each set of homotopy classes, an additional neural network is trained to approximate the Fourier series parameters of the SMMs. The main novelties and contributions of this article are as follows: 1) a closed-form representation of 1-D SMMs is established using Fourier series, which can be accurately learned by neural networks. 2) An intelligent strategy is developed to cluster workspace locations and group their SMMs by homotopy class, greatly improving the accuracy of the approximated SMMs. 3) The SMMs efficiently computed by this new algorithm provide global configuration space information associated with workspace locations, which can be used for real-time global motion planning.

The rest of this article is organized as follows. The next section introduces the background on fault-tolerant motion planning and the nullspace projection method of computing SMMs. In Section IV, a learnable representation of SMMs based on Fourier series is proposed. In Section V, a method to approximate SMMs using neural networks is developed. The performance of this method is validated and its application to real-time fault-tolerant motion planning is discussed in Section VI. Finally, Section VII concludes this article.

II. Related Works

In the context of motion planning, robot tasks can be generally divided into two categories: point-to-point tasks and trajectory following tasks, both represented in the robot's task space and requiring distinct fault-tolerant motion planning strategies. For point-to-point tasks, the reachability of the task point after the failure of one arbitrary joint should be checked at each configuration along a trajectory [7]. The simplest method to guarantee reachability is to constrain the robot's configuration within the ranges of the bounding box enclosing the SMMs belonging to the task point. These SMMs represent all possible joint configurations that reach the desired end-effector location. For multiple task points, fault tolerance can be guaranteed by constraining the robot to stay within the intersection of all the SMM bounding boxes [8]. However, this method relies on very restrictive conditions, such as the existence of the intersection of the bounding boxes. This necessitates an approach to determine the optimal trajectory to reach the overlap of the bounding boxes as soon as possible to maximize the probability of task completion [9]. For trajectory following tasks, the task space trajectory is discretized and the SMMs are computed for each location. The fault-tolerant joint configurations are identified by checking whether or not the remaining trajectory is reachable given the failure of one arbitrary joint [10]. This method requires computing all of the SMMs for each discrete location along the given trajectory, which is very inefficient. An alternative approach is to place the end-effector trajectory inside the fault-tolerant workspace, which is defined as the intersection of the prefailure and postfailure workspaces given a set of artificial joint limits [11].

Once a failure has occurred, some failure recovery strategy for completing the rest of the task is necessary. For point-to-point tasks, either the inverse kinematics [12] or more advanced motion planning algorithms, such as Rapidly-Exploring Random Trees$*$ (RRT$*$) [13], can be used with the robot's postfailure structure to complete the given task. For trajectory following tasks, special care must be taken to ensure that tracking errors after a joint failure are minimized. This problem can be solved as an optimization problem, where joint velocities are minimized subject to the desired end-effector velocity [14], [15].

Because SMMs represent all of the joint configurations to reach a target location, the accurate and efficient computation of SMMs is very important. The most common general method for computing SMMs, named the nullspace projection method, is moving along the null space of the Jacobian using a small step size to reach a new configuration, and this process is repeated until the entire SMM is computed. To keep the end-effector at the given workspace location, the end-effector error is minimized by using the pseudoinverse of the Jacobian [17]. This method may have issues computing SMMs containing singular configurations. An algorithm based on this method is developed in [18] to compute SMMs with high dimensional nullspaces.

An alternative method for computing SMMs is using redundant parameters to reduce a redundant robot to a nonredundant robot, and solving the inverse kinematics with a finite number of solutions. Some examples of redundant parameters include the first joint angle [19], the end-effector orientation angle [20], and the arm angle defined as the angle between the arm plane and the reference plane for a spatial 7R robot [21]. Recently, two search-based algorithms have been developed to compute SMMs. The first computes all SMMs associated with homotopy classes of interest by randomly sampling joint configurations, and driving these configurations to the closest SMM by using Jacobian-based inverse kinematics [22]. The other method discretizes the joint space into a grid, and the SMMs are solved for by searching through the grid elements that are intersected by the null space of their neighbors [23].

As all of the above approaches are iterative approaches, their computational cost is very high, and thus none of them can be used for real-time global motion planning. An early attempt to improve the efficiency of computing SMMs was developed in [24] by using neural networks to approximate the SMMs belonging to distinct workspace regions. This method uses a representation of SMMs that is difficult to learn, and it clusters workspace locations by homotopy classes without accurate data on the number and shape of SMM branches, which makes learning complex SMMs difficult and causes large approximation errors. To solve these problems, a learning-based algorithm is developed to approximate SMMs efficiently and accurately, which can be used in real-time global motion planning.

III. Background

A. Fault-Tolerant Motion Planning Using SMM Bounding Boxes

A common task for a robot is a point-to-point task, where the robot moves from a starting point to a task point with no restrictions on the trajectory it follows. To guarantee fault-tolerance, the robot should be able to reach the task point despite the failure of one arbitrary joint. The SMMs belonging to a workspace location consist of all possible joint configurations to reach this point. Therefore, the reachability of a task point after a failure can be guaranteed by constraining all joints to move within the ranges of the SMMs associated with this task [8]. For a single task point, these ranges are defined as the bounding boxes enclosing the SMMs. For example, Fig. 2(a) shows the SMM of task point a and its associated bounding box in purple. If either joint 2 or joint 3 are locked inside of the SMM bounding box, at the green configuration for instance, the robot can still reach task point a by moving to one of the four yellow configurations. On the contrary, if either joint 2 or joint 3 are locked outside of the SMM bounding box, at the red configuration for instance, then the robot cannot reach task point a. Likewise, the two SMMs belonging to task point b are shown with their respective bounding boxes in red in Fig. 2(b). For multiple task points, these ranges are defined as the intersection of the bounding boxes of each task point, as shown by the green regions. The intersection of SMM bounding boxes provides a set of artificial joint limits for the robot. Various motion planning algorithms can be used to constrain the robot's motion within the artificial joint limits. After a failure occurs, the artificial joint limits are released, and the robot moves toward the configuration along the SMM where the value of the locked joint is valid.

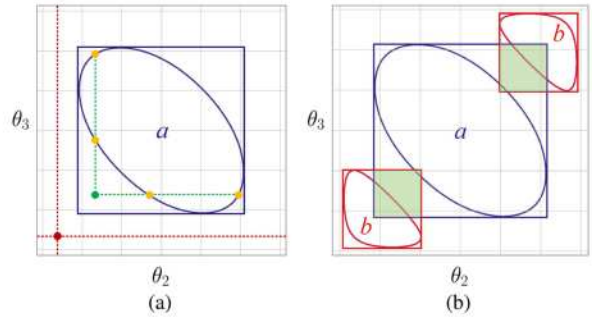

Fig. 2. (a) Bounding box of the SMM belonging to task point a. The green configuration in (a) is fault tolerant because the four yellow configurations along the SMM of task point a can still be reached even after a failure of either joint 2 or joint 3. Conversely, the red configuration in (a) is fault intolerant because none of the configurations along the SMM can be reached after a failure of either joint 2 or joint 3. (b) Bounding box of the two SMMs belonging to task point b and their intersections with the bounding box of the SMM of task point a.

The implementation of this fault-tolerant motion planning algorithm is relatively straightforward. However, computing all of the SMMs of an arbitrary workspace location is difficult and time consuming. In particular, for scenarios requiring real-time fault-tolerant motion planning, the computational efficiency of the method used to compute SMMs is critical.

B. Nullspace Projection Method for Computing SMMs

The most commonly used method for computing SMMs of robots with one degree of redundancy is the nullspace projection method, which is an iterative approach that integrates along the tangent vector of an SMM [17]. This tangent vector is the null vector of the Jacobian $\mathbf{J}$, denoted $\hat{\mathbf{n}}_\mathbf{J}$, which represents joint velocities that map to zero end-effector velocity. Therefore, in each iteration, the change in joint configuration $\Delta\boldsymbol{\theta}$ used to move along the SMM can be computed as follows:

$$\Delta\boldsymbol{\theta} = \gamma\hat{\mathbf{n}}_\mathbf{J} + \mathbf{J}^{+}\Delta\mathbf{x}_e \tag{1}$$

where γ in the first term is the step size along the null vector, and the second term $\mathbf{J}^{+}\Delta\mathbf{x}_e$ represents the end-effector error correction.

The nullspace projection method can compute SMMs accurately. However, the main issue of this method is its low computational efficiency due to the following two properties. First, the computation of each SMM uses an iterative approach where a small step size is used to preserve end-effector accuracy. Second, the number of SMM branches belonging to a workspace location is unknown prior to computation. Thus, to find all SMM branches, the algorithm needs to try many different initial configurations to find potentially new branches. Furthermore, computing these initial configurations using inverse kinematics is also an iterative approach, which further increases computational cost. This motivates the development of a learning-based

SMM approximation algorithm, which utilizes the highly efficient computational characteristics of neural networks, enabling the real-time computation of SMMs.

IV. Learnable Closed-Form Representation of SMMs

A. Representing SMMs as Fourier Series

To effectively approximate SMMs with neural networks, the SMMs should be represented in a manner that maximizes their ability to be learned. Unfortunately, analytical equations for SMMs are unavailable due to their high degree of nonlinearity. Thus, the SMMs computed by most existing methods, including the nullspace projection method, are represented as sequences of discrete joint configurations, which are difficult to learn effectively. This is because the accurate estimatation of SMMs requires a large amount of discrete joint configurations. This corresponds to an increase in neural network complexity and the memory footprint of the training datasets. These issues are evident from the research in [24], where the SMMs of a planar 3R robot were learned with an average forward kinematics (FK) error of about 4% of the robot's total link length, which is very large for such a simple robot.

To solve these issues, a learnable closed-form representation of SMMs is proposed based on the Fourier series of the sequences of discrete joint configurations computed by the nullspace projection method. Using this representation of SMMs in the frequency domain, each SMM can be accurately represented as a continuous function of a finite set of Fourier series components, which can be precisely learned by simple neural networks. For a workspace location $\mathbf{x}$, its associated SMMs can be represented as a set of branches, denoted $\mathcal{M}_{\mathbf{x}} = \{\mathcal{M}_{\mathbf{x}}^{(1)}, \ldots, \mathcal{M}_{\mathbf{x}}^{(B)}\}$, where B is the number of branches. Each branch $\mathcal{M}_{\mathbf{x}}^{(i)}$ is represented by a sequence of discrete joint configurations, denoted $\mathcal{M}_{\mathbf{x}}^{(i)} = \{\boldsymbol{\theta}_{\mathbf{x}}^{(1)}, \ldots, \boldsymbol{\theta}_{\mathbf{x}}^{(N)}\}$, where N is the number of discrete joint configurations. The value of N can be kept constant for all workspace locations and SMM branches through interpolation. A sequence $\mathcal{M}_{\mathbf{x}}^{(i)}$ can be transformed to its corresponding Fourier series using the discrete Fourier transform, defined as follows:

$$\boldsymbol{\Theta}_{\mathbf{x}}^{(k)} = \sum_{i=1}^{N} \boldsymbol{\theta}_{\mathbf{x}}^{(i)} \exp\left(\frac{-j2\pi(k-1)(i-1)}{N}\right) \tag{2}$$

where k represents the k^{th} Fourier series frequency component of the given sequence, and j is the imaginary unit. Thus, an alternative representation of $\mathcal{M}_{\mathbf{x}}^{(i)}$ is its Fourier series representation, denoted $\mathcal{M}_{\mathbf{x}}^{(i)} = \{\boldsymbol{\Theta}_{\mathbf{x}}^{(1)}, \ldots, \boldsymbol{\Theta}_{\mathbf{x}}^{(K)}\}$, where K is the number of Fourier series frequency components, which equals the number of discrete joint configurations N. Once the Fourier series components have been computed, a closed-form equation for the joint configurations along the SMMs, as a function of parameter t, is obtained from the inverse discrete Fourier transform as follows:

$$\boldsymbol{\theta}_{\mathbf{x}}(t) = \frac{1}{K} \sum_{k=1}^{K} \boldsymbol{\Theta}_{\mathbf{x}}^{(k)} \exp\left(\frac{j2\pi(k-1)t}{K}\right). \tag{3}$$

B. Establishing a Unique Representation of SMMs

The above derivation of the Fourier series representation of SMMs is based on the sequences of discrete joint configurations computed by the nullspace projection method. However, the nullspace projection method can choose an arbitrary starting configuration and follow an arbitrary direction along the SMM. Thus, the same SMM can be represented by different sequences of discrete joint configurations, which leads to multiple different Fourier series. To establish a unique representation of SMMs, the following two issues will be addressed in this section: 1) determining a consistent starting configuration, and 2) choosing a unique direction to move along the SMM.

The shift theorem states that changing the first element in a sequence will affect the phase, but not magnitude, of the sequence's Fourier series frequency components. Thus, a method must be established to choose the first joint configuration of a sequence of discrete joint configurations representing an SMM. A consistent choice can be obtained by placing a constraint on the phase of the first positive frequency component of the first joint, then using the shift theorem to adjust the remaining frequencies accordingly. Thus, the shift constant can be computed as follows:

$$\Delta k = \left\lfloor \operatorname{atan2}(\Im(\Theta_{\mathbf{x},1}^{(2)}), \Re(\Theta_{\mathbf{x},1}^{(2)})) \cdot \frac{N}{2\pi} \right\rceil \tag{4}$$

where $\lfloor \cdot \rceil$ denotes the rounding operation to the nearest integer. The symbols $\Im(\cdot)$ and $\Re(\cdot)$ represent the imaginary and real parts of a complex number, respectively. The symbol $\Theta_{\mathbf{x},1}^{(2)}$ represents the second term of the Fourier series of joint 1 for an SMM at workspace location $\mathbf{x}$, i.e., the first positive frequency component of joint 1. This shift constant is used to change each Fourier series component from $\boldsymbol{\Theta}_{\mathbf{x}}^{(k)}$ to $\boldsymbol{\Theta}_{\mathbf{x}}^{(k)} \exp(\frac{-j2\pi k\Delta k}{N})$. Conceptually, this represents reordering the underlying sequence of discrete joint configurations representing the SMM as follows:

$$\mathcal{M}_{\mathbf{x}}^{(i)} = \left\{\boldsymbol{\theta}_{\mathbf{x}}^{(\Delta k)}, \ldots, \boldsymbol{\theta}_{\mathbf{x}}^{(N)}, \boldsymbol{\theta}_{\mathbf{x}}^{(1)}, \ldots, \boldsymbol{\theta}_{\mathbf{x}}^{(\Delta k-1)}\right\}. \tag{5}$$

After the initial joint configuration is determined, the SMM can still be moved along in two opposite directions. This results in two sets of Fourier series with the same frequency components in the opposite order. Therefore, at the starting configuration, a unique direction to move along the SMM must be determined. Once this direction is chosen, the following directions at each configuration of the SMM should be consistent with the previous direction to avoid moving back and forth. As introduced in Section III-B, the direction to move along the SMM, i.e., the tangent vector of the SMM, is the null vector of the Jacobian. The following closed-form equation for the null vector provides smoothly varying directions [25]

$$\hat{n}_{\mathbf{J},i} = (-1^i) \cdot \det(\mathbf{J} \backslash \mathbf{j}_i) \tag{6}$$

where $\hat{n}_{\mathbf{J},i}$ represents the i^{th} element of the null vector and $\mathbf{J} \backslash \mathbf{j}_i$ represents the Jacobian $\mathbf{J}$ with the i^{th} column $\mathbf{j}_i$ removed.

C. Reducing High Frequency Terms of Fourier Series

While the above method based on Fourier series can be used to create a unique representation of SMMs, the number of Fourier

series components, which is equal to the number of discrete joint configurations, can be very large. This large amount of data will be difficult to learn. However, there are some high frequency components that can be removed without losing information of the original SMMs. It has been shown that when Fourier series are applied to periodic sequences that have small and smooth variance between adjacent elements, their high frequency terms are negligible. The research in [17] shows that 1-D SMMs can always be deformed into a unit circle, therefore moving along a fixed direction will always return to an initial configuration, which results in the SMM being periodic. Thus, the following two strategies are taken to ensure the sequences of discrete joint configurations representing SMMs satisfy these properties.

The first strategy is to keep the value of γ in (1) constant and small. This will ensure that the sequences of discrete joint configurations vary smoothly with a small difference. The second strategy is to map joint configurations from the real space to the circle group, denoted $\mathbb{S}^1 = \{z \in \mathbb{C} : |z| = 1\}$, which is defined as follows:

$$\mathcal{T}(\boldsymbol{\theta}) = \exp(j\boldsymbol{\theta}). \tag{7}$$

The circle group is the most accurate representation of joint configurations, as it reflects the geometry of rotational joints, i.e., a rotation of $2\pi n, n \in \mathbb{Z}$, returns to the original joint angle. This can ensure that sequences of discrete joint configurations along an SMM are periodic, regardless of any rotations of $2\pi n$ about a joint. Finally, a low-pass filter can be used to remove the negligible high frequency terms.

Based on all of the above discussion, the final learnable, closed-form representation of SMMs based on the discrete Fourier transform is defined as follows:

$$\boldsymbol{\Theta}_{\mathbf{x}}^{(k)} = \sum_{i=1}^{N} \mathcal{T}(\boldsymbol{\theta}_{\mathbf{x}}^{(i)}) \exp\left(\frac{-j2\pi(k-1)(i-1)}{N}\right). \tag{8}$$

Note that $\mathcal{T}(\boldsymbol{\theta}_{\mathbf{x}}^{(i)})$ is a complex number, but this does not cause any issues for the discrete Fourier transform. The following inverse of $\mathcal{T}$

$$\mathcal{T}^{-1}(\boldsymbol{\theta}) = \text{atan2}(\Im(\boldsymbol{\theta}), \Re(\boldsymbol{\theta})) \tag{9}$$

can return joint configurations from the circle group to the real space. The following equation can be used to map from the final representation of SMMs in (8) back to real joint configurations along SMMs

$$\boldsymbol{\theta}_{\mathbf{x}}(t) = \mathcal{T}^{-1}\left(\frac{1}{K}\sum_{k=1}^{K} \boldsymbol{\Theta}_{\mathbf{x}}^{(k)} \exp\left(\frac{j2\pi(k-1)t}{K}\right)\right). \tag{10}$$

To show the effectiveness of this proposed learnable closed-form representation of SMMs based on Fourier series, SMMs belonging to 100 randomly selected workspace locations for an equal link length planar 3R positioning robot, the optimally fault-tolerant spatial 4R positioning robot designed in [26], and the optimally fault-tolerant spatial 7R positioning and orienting robot designed in [27] are computed. These SMMs are then converted to their Fourier series representation using (8), and different numbers of high frequency components are removed using low-pass filters. After the low-pass filter is applied, the SMMs are converted back to the time domain using (10). The mean absolute error between the filtered and unfiltered SMMs is then calculated, as shown in Fig. 3. It can be seen that using only 40 of the original 128 Fourier series frequency components can achieve mean absolute errors of less than 0.0006, 0.0008, and 0.006 radians for the planar 3R, spatial 4R, and spatial 7R robots, respectively. This extremely small error after reducing the number of variables to 32% of the original data size demonstrates the effectiveness of this representation.

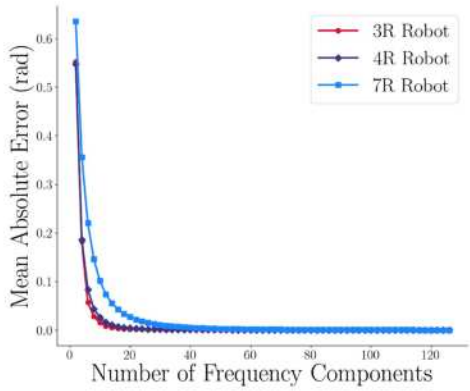

Fig. 3. Mean absolute errors between SMMs of 100 randomly selected workspace locations before and after applying low-pass filters to remove different numbers of frequency components are shown for a planar 3R, a spatial 4R, and a spatial 7R robot.

An illustrative example of representing SMMs using Fourier series is given in Fig. 4 for a planar 3R robot with equal link lengths. The two SMM branches for the given workspace location $[0.30, 0.00]$ are shown in Fig. 4(a), and the pink branch is used to illustrate the method introduced in this section. The sequences of discrete joint configurations of this branch computed by the nullspace projection method are shown in Fig. 4(b), where the individual joints are shown in different colors. These joint configurations are mapped to the circle group as shown in Fig. 4(c). The discrete Fourier transform is applied to the sequence of joint configurations in the circle group. The magnitudes of the Fourier series components of joint 1 are shown in Fig. 4(d), where the gray region represents the removed high frequencies. The phases of the Fourier series components of joint 1 are shown in Fig. 4(e). It can be seen that after applying the shift theorem, the phase of the first positive frequency component ($k = 2$) of joint 1 is roughly zero, as shown by the blue point.

V. Predicting SMM Homotopy Classes and Respective Fourier Series

A. Algorithm Overview

Now that a learnable representation for SMMs has been developed, this section will employ deep feedforward neural networks to efficiently and accurately approximate SMMs for arbitrary workspace locations. This cannot be accomplished with a single neural network because the changes in SMMs with respect to changes in their associated workspace locations are extremely complicated in terms of the number of SMM branches and their shapes. To model these changes, a cellular automaton

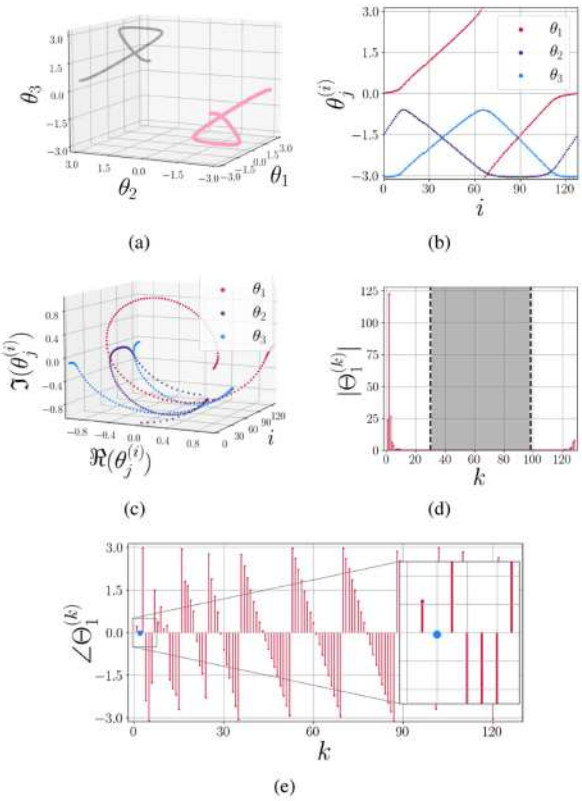

Fig. 4. Example of applying the Fourier series to SMMs is given. (a) Two SMM branches of a given workspace location. For the pink SMM branch, the corresponding sequence of discrete joint configurations represented in the (b) real space and (c) circle group. (d) Magnitudes and (e) phases of the Fourier series components of joint 1.

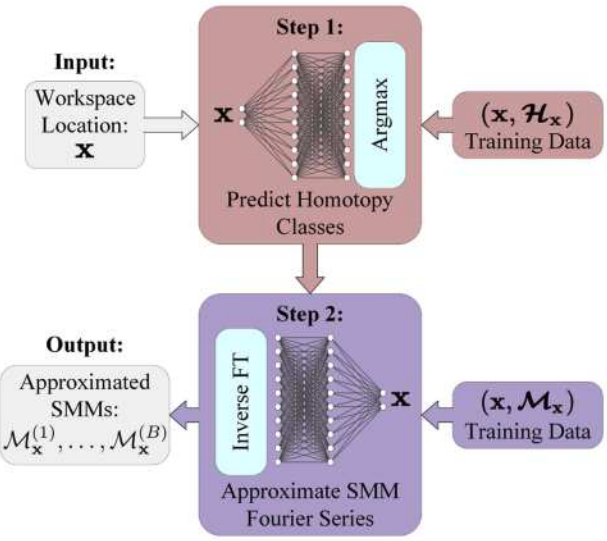

Fig. 5. Algorithm developed to approximate SMMs of an arbitrary workspace location x is shown.

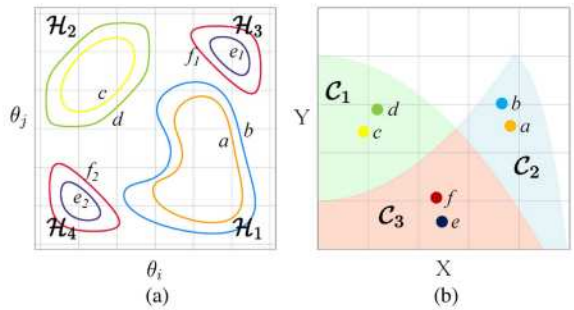

Fig. 6. Example of SMM homotopy classes and workspace location clusters is given. (a) Four homotopy classes determined based on the shape of the given SMM branches. (b) Corresponding workspace locations are grouped into three clusters based on their SMM homotopy classes.

is used to cluster workspace locations based on their number of SMM branches and the shape of these branches. Within each workspace location cluster, SMM branches with similar shapes are grouped together into homotopy classes so they can be learned by the same neural network.

Once the relationships established by the cellular automaton have been learned, the SMMs of a given workspace location are approximated as follows. First, a neural network is used to predict the workspace location cluster and its associated SMM homotopy classes for a given workspace location, as shown in Step 1 in Fig. 5. Second, an additional neural network corresponding to the predicted SMM homotopy classes is used to approximate the SMM branches of the given workspace location, which is Step 2 in Fig. 5. The output of this neural network is the approximated Fourier series for all SMM branches associated with the given workspace location. The joint configurations belonging to these SMM branches are then obtained by using the inverse Fourier transform as defined in (10). Finally, the FK error of the joint configurations along the approximated SMMs can be further reduced by applying Jacobian-based inverse kinematics if high precision is required.

B. Clustering Workspace Locations

As previously discussed, the relationship between SMM branches belonging to different workspace locations can be extremely complicated. For example, in Fig. 6(a) the SMMs of workspace locations a and b have one branch each, and these two SMMs can be smoothly deformed between each other. Thus, they can be classified into the same homotopy class, which is also true for the SMMs of workspace locations c and d. However, the SMMs of workspace locations a and c are not smoothly deformable, and therefore belong to different homotopy classes $\mathcal{H}_1$ and $\mathcal{H}_2$, respectively. Clearly, the SMMs of workspace locations with two branches, such as the SMMs of e and f, do not belong to either $\mathcal{H}_1$ or $\mathcal{H}_2$. The SMM branches e_1 and f_1 belong to the same homotopy class $\mathcal{H}_3$ because they can be smoothly deformed between each other. The same holds for the SMM branches e_2 and f_2, which belong to homotopy class $\mathcal{H}_4$. Based on the above analysis, the workspace locations a, b, c, d, e, and f can be grouped into three clusters $\mathcal{C}_1$, $\mathcal{C}_2$, and $\mathcal{C}_3$, as shown in Fig. 6(b), each having SMMs of the same homotopy classes.

Cellular automata are discrete computational models that use a collection of stateful cells that evolve as a function of some update rule and the states of each cell's neighbors. Clustering the workspace locations based on the features of the SMMs is

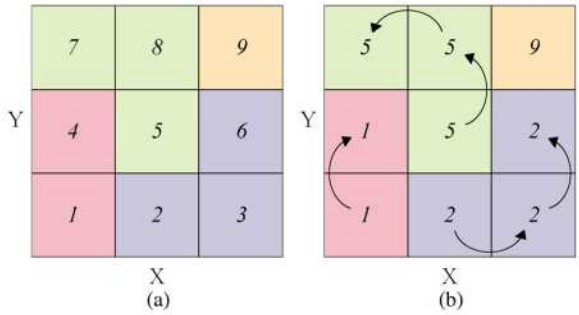

Fig. 7. Example of clustering workspace locations using cellular automata is given. Workspace locations with the same number of SMM branches are shown in the same color. (a) and (b) Initial states of the cells and the final states after convergence, respectively.

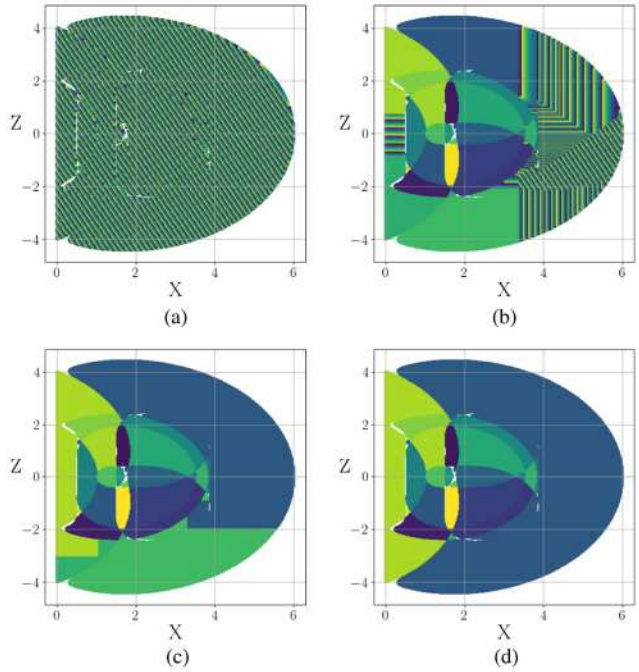

Fig. 8. Real example of applying cellular automata to cluster workspace locations of the spatial 4R positioning robot designed in [26] is given. The states of the cells after (a) 0% , (b) 33% , (c) 67% , and (d) 100% of the evolutionary process are shown. Finally, this results in 24 workspace location clusters, shown by the different colors in (d).

extremely complicated because of the following reasons. First, the shape of each workspace location cluster is highly irregular. Second, workspace locations whose SMMs contain singular configurations act as a boundary between workspace location clusters, and these boundaries are very difficult to accurately identify. Cellular automata can easily handle both of these problems by using a carefully constructed workspace grid and an appropriate update rule that only allows for cells within the same workspace location cluster to communicate with each other. Therefore, cellular automata are applied in this work to achieve the following two goals: 1) to cluster workspace locations based on their number of SMM branches, and 2) to group the SMMs of workspace locations within the same workspace location cluster by homotopy classes based on their shapes.

To achieve the first goal, the workspace locations of a given robot arm are divided into a grid. A unique integer is assigned to each cell, which represents its state. The neighbors of a cell are defined as its adjacent cells with the same number of associated SMM branches. During each step, the state of each cell updates to become the minimum state amongst itself and its neighbors. Finally, the evolution process converges when each cell has the same state as its neighbors, and cells with the same state represent a workspace location cluster. A simple illustrative example is given in Fig. 7, where the workspace is divided into a grid with nine cells. Cells with the same number of SMM branches are shown in the same color. The integers in Fig. 7(a) are the cells' initial states. After two steps, the cellular automaton converges, with the final states shown in Fig. 7(b).

However, workspace locations with the same number of SMM branches may not belong to the same workspace location cluster because their SMMs are quite different in shape and thus belong to different homotopy classes, such as workspace locations a and c, in Fig. 6. As SMMs with singular configurations act as a boundary between homotopy classes, their corresponding workspace locations will bound workspace location clusters. Therefore, these workspace locations will be identified and used to separate cells in the cellular automaton with the same number of SMM branches.

To improve the computational efficiency of the cellular automata, the dimension of its positioning workspace grid is reduced by one. This can be done because rotating about joint 1 will shift the SMMs along the θ_1 axis, but will not change their shape. Thus, the 3-D positioning workspace grid can be fully represented by a half-plane whose normal is perpendicular to the rotation axis of joint 1 [11]. Similarly, for spatial 7R positioning and orienting robots, the dimension of the orienting workspace grid can also be reduced by one if the last rotational joint does not affect its end-effector position.

A real example of using cellular automata to cluster the workspace locations of the optimally fault-tolerant spatial 4R positioning robot designed in [26] is given in Fig. 8. The 3-D workspace grid fully represented by the X–Z half-plane. The states of each cell after 0% , 33% , 67% , and 100% of the evolutionary process are shown in Fig. 8(a), (b)(c), and (d), respectively. Finally, the workspace locations are grouped into 24 clusters, which are shown by different colors in Fig. 8(d). Workspace locations whose SMMs contain singular configurations are seen as the white points separating different workspace location clusters.

C. Grouping SMMs by Homotopy Classes

After clustering workspace locations, the cellular automaton is used again to group SMMs of each workspace location cluster by homotopy class, such that they can be accurately learned by a single neural network. To accomplish this goal, the deformation matrix is proposed to determine the pairs of SMM branches, associated with two workspace locations, that belong to the same homotopy class. Given workspace locations x_1 and x_2 belonging to the same workspace location cluster with B SMM branches,

the deformation matrix $\mathbf{D}$ is defined as

$$\mathbf{D} = \begin{bmatrix} \Delta(\mathcal{M}_{\mathbf{x}_1}^{(1)}, \mathcal{M}_{\mathbf{x}_2}^{(1)}) & \dots & \Delta(\mathcal{M}_{\mathbf{x}_1}^{(1)}, \mathcal{M}_{\mathbf{x}_2}^{(B)}) \\ \vdots & \ddots & \vdots \\ \Delta(\mathcal{M}_{\mathbf{x}_1}^{(B)}, \mathcal{M}_{\mathbf{x}_2}^{(1)}) & \dots & \Delta(\mathcal{M}_{\mathbf{x}_1}^{(B)}, \mathcal{M}_{\mathbf{x}_2}^{(B)}) \end{bmatrix} \tag{11}$$

where $\Delta(\mathcal{M}_{\mathbf{x}_1}^{(i)}, \mathcal{M}_{\mathbf{x}_2}^{(j)})$ represents the deformation between the i^{th} branch of workspace location $\mathbf{x}_1$ and the j^{th} branch of workspace location $\mathbf{x}_2$. This metric is defined as the average of the distances from each configuration along $\mathcal{M}_{\mathbf{x}_1}$ to its closest configuration on $\mathcal{M}_{\mathbf{x}_2}$, i.e.,

$$\Delta(\mathcal{M}_{\mathbf{x}_1}, \mathcal{M}_{\mathbf{x}_2}) = \frac{1}{N} \sum_{i=1}^{N} \min_{\boldsymbol{\theta}_{\mathbf{x}_2}^{(j)} \in \mathcal{M}_{\mathbf{x}_2}} \{\delta(\boldsymbol{\theta}_{\mathbf{x}_1}^{(i)}, \boldsymbol{\theta}_{\mathbf{x}_2}^{(j)})\}. \tag{12}$$

The function δ represents the distance between joint configurations $\boldsymbol{\theta}^{(a)}$ and $\boldsymbol{\theta}^{(b)}$ represented in the circle group, and is defined as follows:

$$\delta(\boldsymbol{\theta}^{(a)}, \boldsymbol{\theta}^{(b)}) = \sqrt{\sum_{i=1}^{n} \mathcal{T}^{-1} \left(\frac{\theta_i^{(a)}}{\theta_i^{(b)}} \right)^2}. \tag{13}$$

If $\Delta(\mathcal{M}_{\mathbf{x}_1}^{(i)}, \mathcal{M}_{\mathbf{x}_2}^{(j)})$ is very small, then these two branches can be smoothly deformed between each other, and thus belong to the same homotopy class.

To pair the B SMM branches of workspace locations $\mathbf{x}_1$ and $\mathbf{x}_2$ in an optimal way, the total deformation between all of the paired SMM branches should be minimized. In addition, each SMM branch belongs to only one homotopy class, and different SMM branches of the same workspace location should belong to different homotopy classes. This can be viewed as the assignment problem, where the SMM branches of $\mathbf{x}_2$ are "assigned" to the SMM branches of $\mathbf{x}_1$, with the cost between them being their respective deformation. The Hungarian algorithm is an efficient method for solving this problem, where the above optimization problem is equivalent to minimizing the trace of $\mathbf{D}$ with respect to its column permutations, i.e.,

$$\mathbf{P}^* = \underset{\mathbf{P}}{\operatorname{argmin}} \operatorname{tr}(\mathbf{DP}) \tag{14}$$

where $\operatorname{tr}(\cdot)$ is the trace operator and $\mathbf{P}$ is an arbitrary column permutation matrix. Conceptually, the solution $\mathbf{P}^*$ produces a set of paired SMM branches between the two workspace locations, $\{(\mathcal{M}_{\mathbf{x}_1}^{(i)}, \mathcal{M}_{\mathbf{x}_2}^{(j)}) : p_{i,j}^* = 1\}$, which belong to the same homotopy class. For example, given workspace locations e and f in Fig. 6, SMM branch e_1 should be paired with f_1, and e_2 with f_2.

To apply this process to the entire workspace, the cellular automaton is used again with the same evolution mechanisms as described in the previous section, but with a different definition of state. The state in this problem is defined as a binary value that represents whether or not the SMM branches of a given workspace location have been paired with the SMM branches of another workspace location belonging to the same workspace location cluster. After the workspace clustering cellular automaton converges, the states of the cells whose initial states were the minimum in their respective workspace location cluster are now set to one. The states of the remaining cells are set to zero. During each step, each cell has their SMM branches paired with the SMM branches of their neighboring cells with a state of one. Once the SMMs branches of a cell have been paired, the cell's state is updated to one. This cellular automaton converges when every cell has a state of one, and it identifies the corresponding SMM homotopy classes for each workspace location cluster.

D. Predicting Homotopy Classes and Approximating SMM Fourier Series

With the workspace locations clustered and their SMM branches grouped by homotopy class, the neural networks are now implemented to approximate the SMMs of arbitrary workspace locations. First, a deep feedforward neural network is used to predict the probabilities that the input workspace location belongs to each workspace location cluster. The argmax function is used to select the most likely workspace location cluster and thus identify the homotopy classes associated with this cluster. For each set of homotopy classes, another neural network will approximate the real and imaginary parts of the Fourier series of the SMMs belonging to this given workspace location.

To learn the probabilities of a workspace location belonging to each workspace location cluster, the first neural network is trained on the one-hot encodings of the labeled workspace location clusters. This neural network is trained using backpropagation to minimize the following cross entropy loss function

$$\mathcal{L}(\mathbf{Y}, \widehat{\mathbf{Y}}) = \frac{1}{D} \sum_{i=1}^{D} \sum_{c=1}^{C} -y_c^{(i)} \cdot \log \left(\operatorname{softmax} \left(\hat{\mathbf{y}}^{(i)}, c \right) \right) \tag{15}$$

where $\mathbf{Y}$ is the training set of one-hot encoded cluster labels of the workspace locations and $\widehat{\mathbf{Y}}$ is the set of probabilities that each workspace location belongs to a given cluster. D is the size of the given training set and C is the total number of workspace location clusters. The softmax function with inputs $\mathbf{y} \in \mathbb{R}^k$ and an integer i $(1 \leq i \leq k)$ is defined as follows:

$$\operatorname{softmax}(\mathbf{y}, i) = \frac{\exp(y_i)}{\sum_{j=1}^{k} \exp(y_j)}. \tag{16}$$

To learn the Fourier series of SMMs, the neural networks corresponding to each set of homotopy classes are trained on the real and imaginary parts of the Fourier series components of the SMMs. All of the SMM approximation neural networks are trained using backpropagation to minimize the following mean squared error loss function

$$\mathcal{L}(\mathbf{Y}, \widehat{\mathbf{Y}}) = \frac{1}{D} \sum_{i=1}^{D} \|\mathbf{y}^{(i)} - \hat{\mathbf{y}}^{(i)}\|_2^2 \tag{17}$$

where $\mathbf{Y}$ is the set of Fourier series components representing the SMMs belonging to each workspace location, $\widehat{\mathbf{Y}}$ is the approximated Fourier series components, and D is the size of the given training set.

E. Further Reducing FK Error of Approximated SMMs

In some applications, such as human–robot handover tasks, picking items out of cluttered objects, and placing items in containers, the required precision of the end-effector can be

TABLE I
DH PARAMETERS OF THE OPTIMALLY FAULT TOLERANT SPATIAL 4R ROBOT

i	α_i [rad]	a_i [m]	d_i [m]	θ_i [rad]
1	$\pi/2$	1.4142	0	0
2	$-\pi/2$	1.4142	1	0
3	$\pi/2$	1.4142	-1	0
4	0	1.2247	0.5	0

TABLE II
DH PARAMETERS OF THE OPTIMALLY FAULT TOLERANT SPATIAL 7R ROBOT

i	α_i [rad]	a_i [m]	d_i [m]	θ_i [rad]
1	-1.7104	0.17	0	0
2	-1.9897	1.42	1.67	0
3	-1.1519	1.42	-0.69	0
4	0.8726	0.56	-1.77	0
5	-1.6057	1.32	2.42	0
6	-1.6150	1.27	-0.38	0
7	0	0	0.95	0

relatively low. The approximated SMMs should be accurate enough to be used in these cases. However, some applications may rely on high-precision operations requiring extremely small FK errors, e.g., assembly tasks or surgery. In these use cases, if the FK error of the approximated joint configurations is larger than the application-specific tolerance, it can be further reduced by using Jacobian-based inverse kinematics, where the joint configurations are updated as follows:

$$\Delta\theta = \mathbf{J}^{+}\Delta\mathbf{x}_e. \tag{18}$$

This very simple error correction mechanism is widely applied, as seen in the nullspace projection method from (1), in the sampling-based method in [22], and in the learning-based inverse kinematics method in [28]. Because the approximated SMMs computed by the proposed method have relatively low errors, only a very small number of iterations of the above Jacobian-based inverse kinematics are required to achieve extremely high precision. The small addition to the computational cost does not have any noticeable impact on the real-time applicability of the proposed method.

VI. RESULTS

A. SMMs Estimation

The above proposed method is validated using the following three robots: an equal link length planar 3R positioning robot, the optimally fault-tolerant spatial 4R positioning robot designed in [26], and the optimally fault-tolerant spatial 7R positioning and orienting robot designed in [27]. These two optimally fault-tolerant robots are obtained from their respective optimally fault-tolerant Jacobians, which are defined as being isotropic before a joint failure and having the maximum worst-case failure tolerance after an arbitrary joint failure. While these robots are optimized using local fault-tolerance metrics (i.e., prefailure and postfailure dexterity), their global fault-tolerant performance (i.e., reachability of a target point after an arbitrary joint failure) cannot be guaranteed without using SMM information. The Denavit–Hartenberg parameters for the 4R and 7R robots are listed in Tables I and II, respectively. It can be seen that these two robots have much more complicated structures than many standard spatial 4R and spatial 7R industrial robots, and thus their SMMs can be extremely complicated. To generate the training data using the cellular automata, the entire positioning workspace is discretized into 200 and 100 000 (250×400) for the 3R and 4R robots, respectively. Roughly 10% of the positioning workspace for the 7R robot is discretized into 12 800 (80×160) cells, while the entire orienting workspace associated with these positions is discretized into 12 000 cells. The discrete joint configurations along the SMMs computed by the nullspace projection method were interpolated to have 128 samples, and only the 90 lowest frequencies were kept once converted to their Fourier series representation.

The structures and parameters of the implemented neural networks are given as follows. The neural networks used to predict the workspace location cluster probabilities have 2, 3, and 4 hidden layers with 5, 50, and 150 neurons each for the 3R, 4R, and 7R robots, respectively. The activation function for these neural networks is the Leaky Rectified Linear Unit (ReLU) function, and they are trained using the AdamW [29], [30] optimizer with a learning rate of 0.001. The neural networks used to predict the Fourier series of SMM branches have 4, 5, and 6 hidden layers with 150 neurons each for the 3R, 4R, and 7R robots, respectively. The activation function for these networks is the Leaky ReLU function, and they are also trained using the AdamW optimizer, but with a learning rate of 0.003.

To prove the performance of the proposed method, the trained neural networks belonging to each robot are used to approximate the SMMs of 10 000 randomly sampled workspace locations. These approximations are compared with the SMMs computed by the nullspace projection method, the sampling-based method in [22], and the grid-based method in [23], both in terms of efficiency and accuracy. To compare the computational efficiency, all of the algorithms are implemented on the same computer, which has an Intel i7-8565 U CPU, 16 GB of RAM, an Nvidia MX150 GPU, and 3 GB of VRAM; and the runtime of all of the methods are recorded.

To validate the accuracy of the proposed method, three metrics are used. The first metric considers the FK error of a single joint configuration along the approximated SMM, which is the distance between the workspace position corresponding to this approximated joint configuration, $\hat{\mathbf{p}}$, and the actual workspace position, $\mathbf{p}$. To normalize the FK position error between different robots, the position error, e_p, is divided by the total link length of the arm, L. The FK orientation error, e_o, is calculated as $\arccos((\text{tr}(\mathbf{R}^{\mathrm{T}}\widehat{\mathbf{R}}) - 1)/2)$, where $\mathbf{R}$ is the rotation matrix of the actual orientation and $\widehat{\mathbf{R}}$ is the rotation matrix corresponding to the approximated joint configuration. This metric represents the magnitude of the smallest rotation to make $\mathbf{R}$ and $\widehat{\mathbf{R}}$ equivalent. To combine the position and orientation error for the 7R positioning and orienting robot, the above orientation error is divided by the maximum value of $\arccos$, π, and the total FK error, e, is then the average of the position and orientation error,

TABLE III
COMPARISONS BETWEEN THE PROPOSED METHOD, THE NULLSPACE PROJECTION METHOD, THE SAMPLING-BASED METHOD, AND THE GRID-BASED METHOD

Robot	Algorithm	Time (s)	FK Error (%)	Locations with Different Branch Count
3R	Nullspace Projection	0.63	0.22 ± 0.11	-
	Sampling-Based Method	0.45	0.0095 ± 0.0003	No branch information is provided by this method.
	Grid-Based Method	16.28	2.32 ± 1.34	0.0%
	Proposed Before Error Correction	0.0014	0.29 ± 0.15	0.70%
	Proposed After Error Correction	0.0026	$7.3\times10^{-5} \pm 0.0009$	0.70%
4R	Nullspace Projection	2.31	0.24 ± 0.13	-
	Sampling-Based Method	0.98	0.022 ± 0.54	No branch information is provided by this method.
	Proposed Before Error Correction	0.0016	0.66 ± 0.61	2.16%
	Proposed After Error Correction	0.0047	0.0011 ± 0.016	2.16%
7R	Nullspace Projection	6.82	0.09 ± 0.03	-
	Sampling-Based Method	5.11	0.154 ± 0.855	No branch information is provided by this method.
	Proposed Before Error Correction	0.0017	2.07 ± 1.37	6.02%
	Proposed After Error Correction	0.0083	0.016 ± 0.090	6.02%

defined as follows:

$$e = \frac{1}{2}(e_p + e_o) = \frac{1}{2L}\|\mathrm{p} - \hat{\mathrm{p}}\| + \frac{1}{2\pi}\arccos((\mathrm{tr}(\mathbf{R}^{\mathrm{T}}\widehat{\mathbf{R}}) - 1)/2). \quad (19)$$

Because both e_p and e_o represent percentage errors, the average of these values is used so that e is a percentage as well. The second metric is the FK error of a workspace location, which is the average FK error of all of the joint configurations along the approximated SMMs of this given workspace location. The third metric is the percentage of workspace locations having different numbers of SMM branches between the proposed and nullspace projection methods.

Table III shows the results of the proposed method, before and after error correction using the Jacobian-based inverse kinematics, compared to the nullspace projection method, the sampling-based method, and the grid-based method for each of the three robots. It can be seen that the average computation time of computing the SMMs of each workspace location using the nullspace projection method improves from 0.63 s, 2.31 s, 6.82 s to only 0.0014 s, 0.0016 s, and 0.0017 s using the proposed method before error correction for the 3R, 4R, and 7R robots, respectively. The runtime of the sampling-based method depends on the number of sampled joint configurations for each workspace location. In this experiment, the SMMs are computed using 2000, 3000, and 5000 random samples for the 3R, 4R, and 7R robots to obtain a good approximation of the SMMs. The average computational time is 0.45 s, 0.98 s, and 5.11 s for the 3R, 4R, and 7R robots. Compared to the proposed method, these computational times are many times larger and the efficiency of this method severely degrades in higher dimensions. The runtime of the grid-based method depends on the resolution of the grids. In this experiment, the resolution along each dimension is 100. The runtime of this method for the planar 3R robot is 16.28 s, which is much more computationally expensive than any of the other tested methods. As such, this method is only tested on the planar 3R robot. The runtime of the proposed method after error correction is 0.0026, 0.0047, and 0.0083 s for the 3R, 4R, and 7R robots, respectively, which demonstrates the small computational cost of the error correction mechanism. These results show that the proposed method significantly improves the computational efficiency of computing SMMs, and it is the most computationally efficient method tested. In addition, unlike the nullspace projection method and the sampling-based method, the computational efficiency of the proposed method remains almost constant as the dimension of the joint space increases. This is true because the complexity of the neural networks used to approximate the SMMs is very similar for each robot. Furthermore, because the proposed method relies on highly parallelizable neural networks, another advantage of this method is that its computational time does not increase as fast as increases in the number of input workspace locations. This is not the case for the nullspace projection method and the sampling-based method, as they rely on complex iterative computations that are not well suited for parallelization. Due to the above advantages of the proposed method, only the proposed method can guarantee that global motion planning based on SMMs can be done in real-time.

In the fourth column of Table III, the average and standard deviation of the FK error of all the joint configurations along the approximated SMMs of the 10 000 randomly sampled workspace locations are shown for each robot. As the nullspace projection method and the sampling-based method include an end-effector error correction term, they always have good accuracy with errors around 0.09% to 0.24% and 0.009% to 0.15% , respectively. The FK error of the grid-based method is greatly impacted by the resolution of the grids, and is on average 2.32% for the 3R robot in this experiment, which is much larger than the FK errors of the nullspace projection method and the sampling-based method. The FK error of the proposed method is only slightly larger than the nullspace projection method for the 3R, 4R, and 7R robots, with averages of 0.29% , 0.66% , and 2.1% , respectively. These errors are well within acceptable ranges for any global motion planning applications that do not require extremely high precision. This is particularly true for global fault-tolerant motion planning applications, where only the ranges of the SMMs need to be computed accurately. It can also be seen that the FK error of the proposed method for the planar 3R robot is much less than the 4% FK error achieved in [24]. For high-precision applications, the average FK error of the proposed method can be further reduced to 7.3×10^{-5}% , 0.0011% , and 0.016% for the 3R, 4R, and 7R robots, respectively, using only three inverse kinematics iterations. These

resulting errors are even much smaller than the errors of all of the other methods, and the computational time is still fast enough to be used in real-time motion planning.

The accuracy of the homotopy classes prediction is also analyzed. It is a fact that the true number of SMM branches belonging to a given workspace location is unknown for general robots. However, it can be assumed that when the number of random initial configurations is large enough, the SMMs computed by the nullspace projection method are roughly equivalent to the true SMMs. Therefore, the final column in Table III, which shows the percentage of the 10 000 randomly sampled workspace locations whose number of SMM branches estimated by the proposed method is different than the number computed by the nullspace projection method, can provide insights into the accuracy of the approximated SMMs. It can be seen that the proposed method generally predicts a number of SMM branches consistent with the nullspace projection method. A fundamental issue with the sampling-based method is that it can only obtain discrete points along the SMMs, and thus the geometric information, such as the number of branches and the range of the SMMs along each joint, is not available. Therefore, this method cannot be used in either online or offline global fault-tolerant motion planning.

The distribution of the FK error for each joint configuration along the approximated SMMs for the planar 3R robot are shown in Fig. 9(a), (c), and (e) for the nullspace projection method, the proposed method before error correction, and the proposed method after error correction, respectively. For the nullspace projection method, all of the 1.59×10^6 joint configurations along the approximated SMMs have errors below 0.6% , with the majority having errors around 0.22% . The FK error of the proposed method before error correction is slightly larger, with an average error of 0.29% . However, after applying three iterations of Jacobian-based inverse kinematics, the error was significantly reduced to 7.3×10^{-5}. The 10 000 randomly sampled workspace locations and their associated average FK errors are shown in color for the nullspace projection method, the proposed method before error correction, and the proposed method after error correction in Fig. 9(b), (d), and (f), respectively. It can be seen that the majority of workspace locations have very small average errors for all three methods. The workspace locations with high average FK error tend to be clustered around the workspace boundaries. This can be attributed to the unusual shapes of the SMMs at the workspace boundaries. The distribution of the FK error of the two remaining robots using the three above methods are shown in Figs. 10 and 11. It can be seen that, while the accuracy of the proposed method is less than the nullspace projection method, the error correction is capable of reducing the FK error of the proposed method to be notably lower than that of the nullspace projection method.

Examples of the SMMs approximated by the proposed method are given. For the 3R and 4R robots, Figs. 12 and 13 show the approximated SMMs belonging to three different workspace locations whose average FK errors are in the 25th, 50th, and 75th percentiles as the red, purple, and blue curves, respectively. The SMMs computed by the nullspace projection method belonging

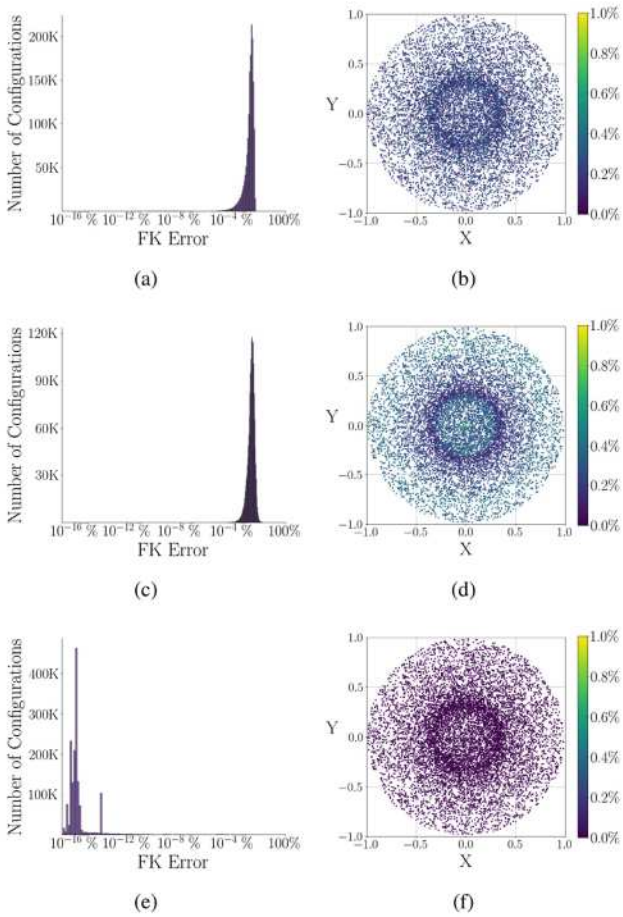

Fig. 9. FK errors of the approximated SMMs are shown for the planar 3R positioning robot. The distribution of FK errors for each joint configuration along the approximated SMMs are shown for the (a) nullspace projection method, (c) proposed method before error correction, and (e) proposed method after error correction. (b), (d), and (f) Average FK errors of the 10 000 randomly sampled workspace locations for each of the above respective methods.

to these same workspace locations are shown underneath these curves as the wider, transparent curves of the same color. As can be seen, the proposed method approximates SMMs, which are almost exactly the same as the SMMs produced by the nullspace projection method, regardless of the complexity of the SMM's shape. For the 7R robot, Fig. 14 shows the approximated SMMs belonging to two different workspace locations whose average FK errors are in the 33rd and 66th percentiles as the red and blue curves, respectively. The SMMs computed by the nullspace projection method are shown as wider, transparent curves of the same color. While the SMMs approximated by the proposed method are slightly different than the SMMs computed by the nullspace projection method, they are still very similar despite the extremely complicated shapes of these SMMs.

B. Real-Time Fault-Tolerant Motion Planning Simulation Experiment

To demonstrate the application of the proposed method in real-time global fault-tolerant motion planning, several examples of locked-joint failures of joints 1 and 4 occurring during the

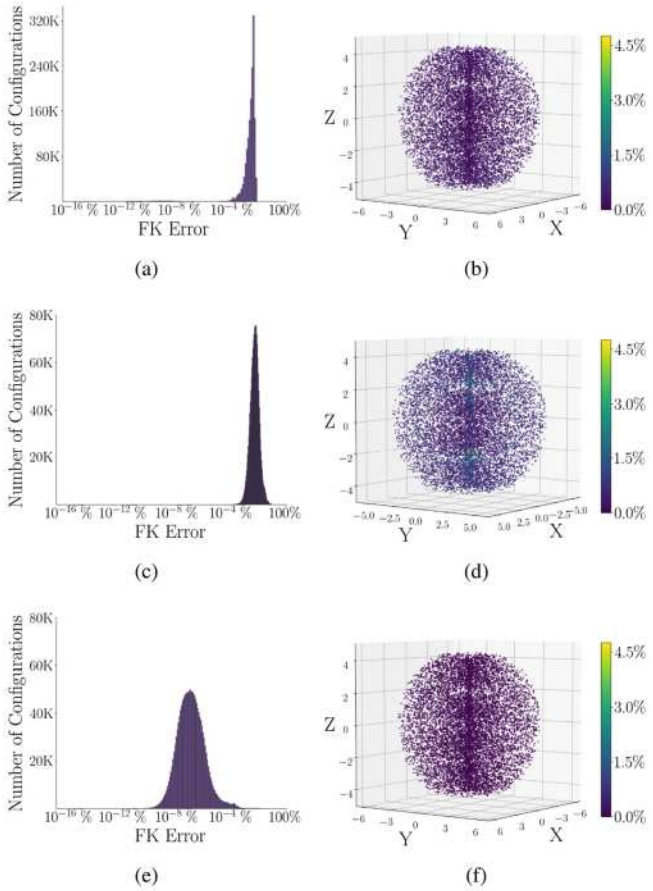

Fig. 10. FK errors of the approximated SMMs are shown for the spatial 4R positioning robot. The distribution of FK errors for each joint configuration along the approximated SMMs are shown for the (a) nullspace projection method, (c) proposed method before error correction, and (e) proposed method after error correction. (b), (d), and (f) Average FK errors of the 10 000 randomly sampled workspace locations for each of the above respective methods.

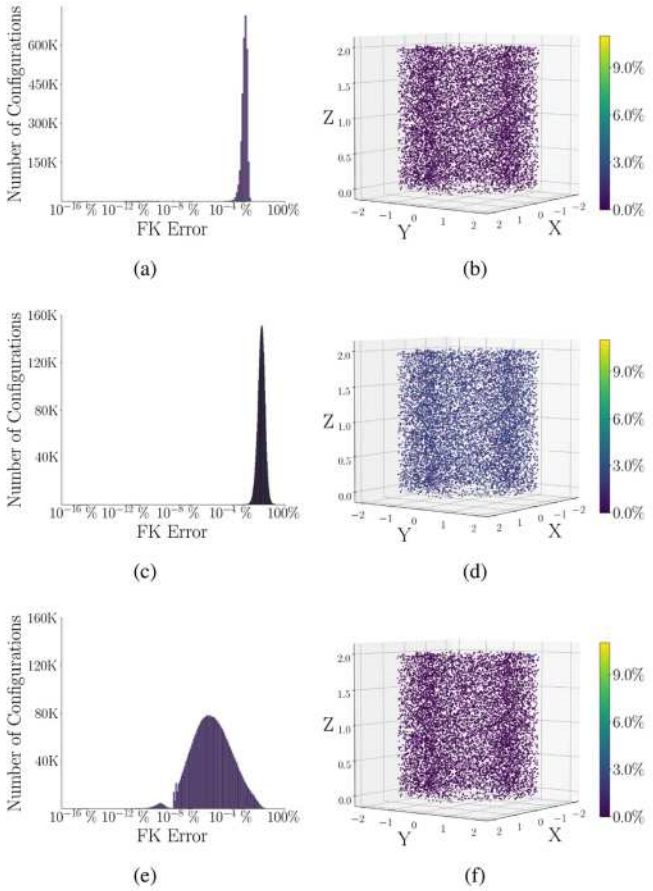

Fig. 11. FK errors of the approximated SMMs are shown for the spatial 7R positioning and orienting robot. The distribution of FK errors for each joint configuration along the approximated SMMs are shown for the (a) nullspace projection method, (c) proposed method before error correction, and (e) proposed method after error correction. (b), (d), and (f) Average FK errors of the 10 000 randomly sampled workspace locations for each of the above respective methods.

operation of the spatial 7R robot are simulated in this subsection. Because the ranges of the SMM bounding boxes along joints 2, 3, 5, 6, and 7 are typically very close to the entire 2π joint range, failures of these joints will not affect the reachability of the task locations and are therefore not considered. Joint limits and self-collisions are not considered in these experiments. First, the robot moves without any information on its next target location, as shown in Fig 15(a). This information is then provided by some external sensing system, such as camera-based object detection and localization, as shown by the green ball (position) with three axes (orientation) in Fig. 15(b). The SMMs associated with the target location are then computed using the proposed method. During the 0.003 s, which the SMMs are being computed, the robot moves toward this target location using the damped least squares solution, as shown by the red trajectory in Fig. 15(c). Note that this red trajectory in Fig. 15(c) is extremely small and almost invisible because only 0.003 s are needed for the proposed method to compute the SMMs. After the SMMs are computed, the robot moves toward the closest bounding box enclosing these SMMs, as shown by the purple trajectory in Fig. 15(d). Because the SMM information is available in real time when using the proposed method, the robot is able to reach and move inside of

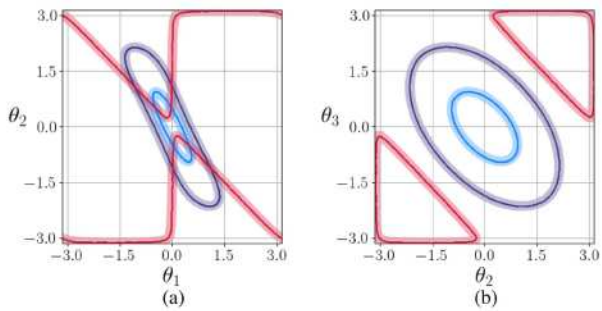

Fig. 12. Examples of the SMMs approximated by the proposed method for three workspace locations of the equal link length planar 3R positioning robot are shown projected onto the (a) θ_1-θ_2 and (b) θ_2-θ_3 planes. The red, purple, and blue curves have errors in the 25th, 50th, and 75th percentiles, respectively. The SMMs computed by the nullspace projection method are shown as the transparent curves of the same color.

the SMM bounding box before a joint failure occurs, as seen by the pink trajectory in Fig. 15(e). One joint is randomly locked during the motion to verify the fault-tolerant performance, where joint one is locked at 5.6 s in this experiment, as seen in Fig. 15(f). The robot is still capable of reaching the target location despite

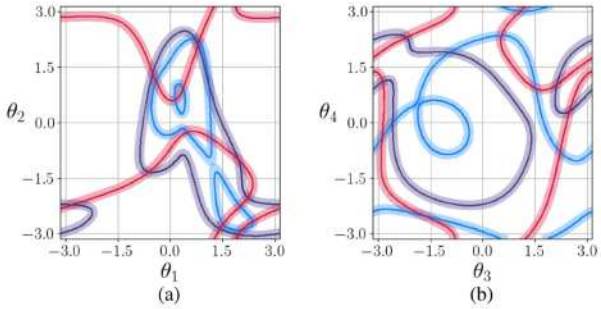

Fig. 13. Examples of the SMMs approximated by the proposed method for three workspace locations of the optimally fault-tolerant spatial 4R positioning robot are shown projected onto the (a) θ_1-θ_2 and (b) θ_3-θ_4 planes. The red, purple, and blue curves have errors in the 25$^{\text{th}}$, 50$^{\text{th}}$, and 75$^{\text{th}}$ percentiles, respectively. The SMMs computed by the nullspace projection method are shown as the transparent curves of the same color.

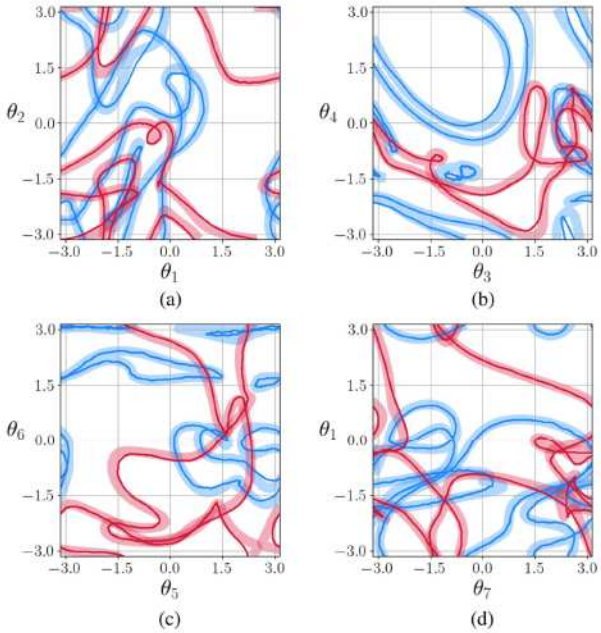

Fig. 14. Examples of the SMMs approximated by the proposed method for two workspace location of the spatial 7R positioning and orienting robot are shown projected onto the (a) θ_1-θ_2, (b) θ_3-θ_4, (c) θ_5-θ_6, (d) and θ_7-θ_1 planes. The red and blue curves have errors in the 33$^{\text{rd}}$ and 66$^{\text{th}}$ percentiles, respectively. The SMMs computed by the nullspace projection method are shown as the transparent curves of the same color.

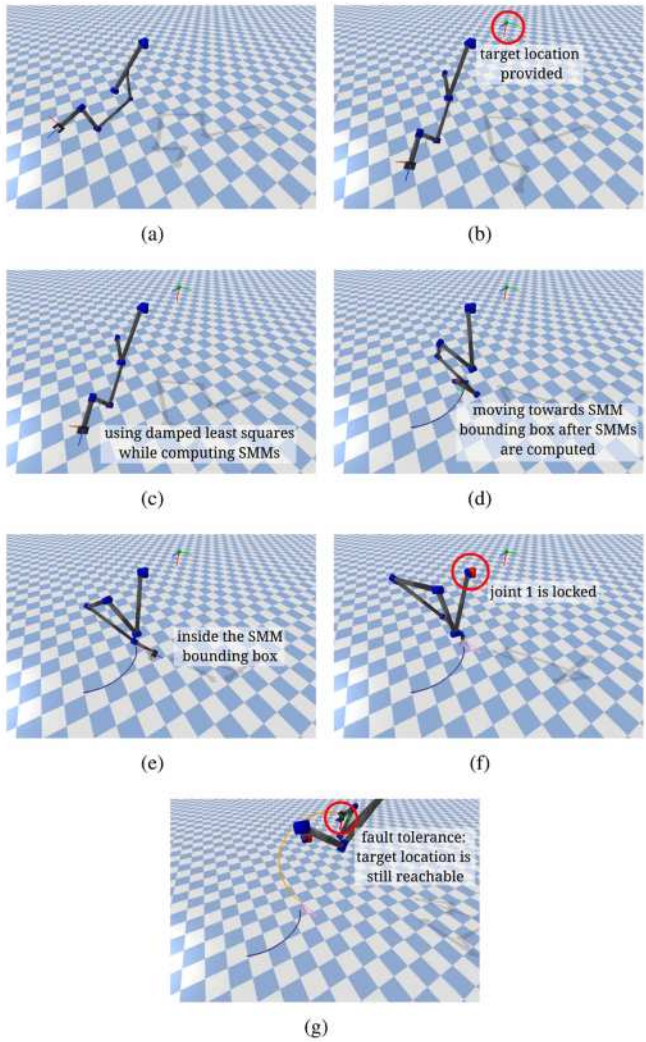

Fig. 15. Example of applying the proposed method to real-time fault-tolerant motion planning is shown. (a) Robot is initially unaware of the next target location. (b) Target location is provided, the SMMs are computed using the proposed method. While they are being computed, (c) robot uses the damped least squares solution to move toward the goal location. After 0.003 s, the SMM information is available and then the (d) robot moves toward the bounding box enclosing the closest SMM. (e) Robot moves inside of the bounding box, (f) joint one is locked at 5.6 seconds. (g) Robot is still able to reach the target location despite the failure of joint one.

the failure of joint one, by following the orange trajectory in Fig. 15(g).

Unfortunately, this fault-tolerant result is not achievable when using the nullspace projection method. The SMMs are computed in 4.3 s, while the robot uses the damped least squares solution to move toward the goal location in the meantime, as shown by the long, red trajectory in Fig. 16(c). After the SMMs are computed, the robot begins to move toward the bounding box enclosing the closest SMM, as shown by the purple trajectory in Fig. 16(d). However, the robot does not have enough time to move into the SMM bounding box before the failure of joint one occurs because the nullspace projection method cannot compute the SMMs fast enough. This is demonstrated by the lack of a pink trajectory, as opposed to the result in Fig. 15(e). Joint one is locked at the same time as above, as shown in Fig. 16(e). Because the robot is not inside of the SMM bounding box, it is not able to reach the goal location after the failure of joint one, as shown in Fig. 16(f). The failure of joint 4 is also simulated and the results demonstrate that the proposed method works equally well when joints other than joint 1 are locked.[1]

C. Real-Time Fault-Tolerant Motion Planning Physical Experiment

To further validate the proposed method in real-time global fault-tolerant motion planning, two physical experiments are conducted on a planar 3R robot which is generated from a

[1] The simulation videos are available on YouTube at: https://youtu.be/orYHjaKYb2Y

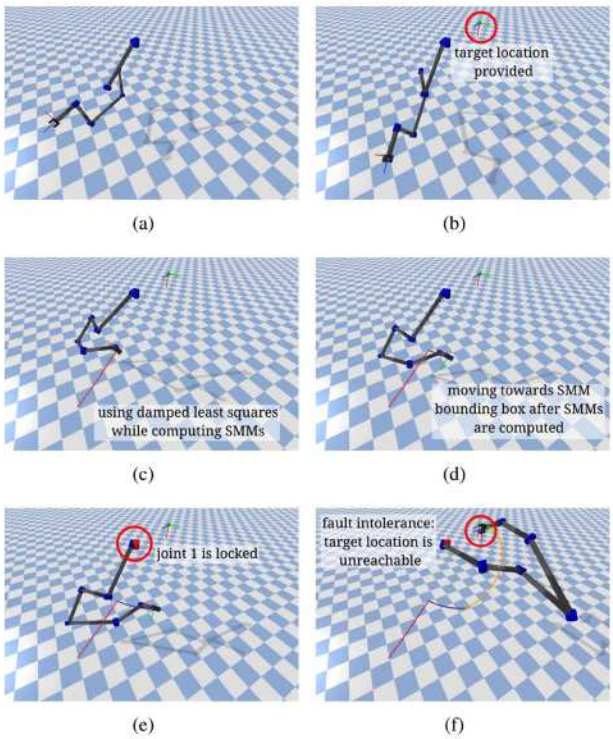

Fig. 16. Comparative example of applying the nullspace projection method to real-time fault-tolerant motion planning is shown. (a) Robot is initially unaware of the next target location. (b) Once the target location is provided, the SMMs are computed using the nullspace projection method. (c) They are being computed, the robot uses the damped least squares solution to move toward the goal location. (d) After 4.3 s, the SMM information is available and then the robot moves toward the bounding box enclosing the closest SMM. (e) Because the nullspace projection method cannot compute the SMMs fast enough, the robot is not able to reach the SMM bounding box before joint one is locked at 5.6 s. (f) Robot is unable to reach the target location after the failure of joint one.

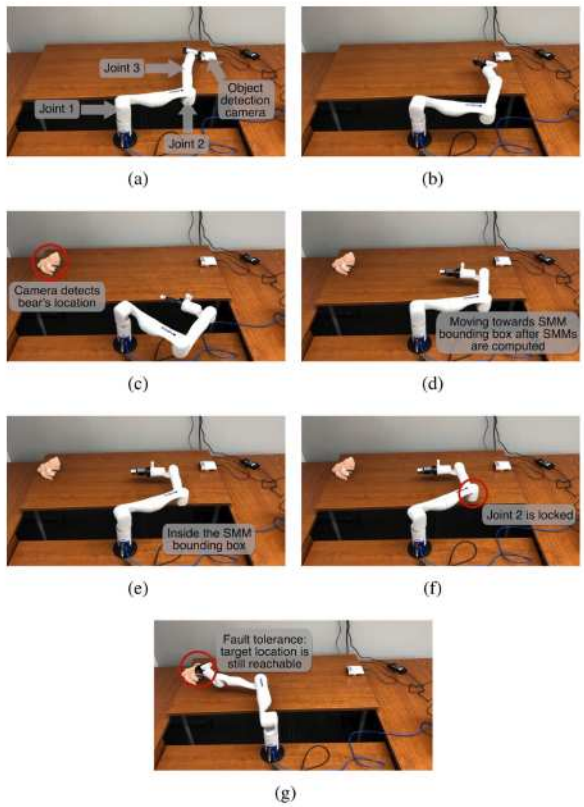

Fig. 17. (a) Physical experiment of real-time global fault-tolerant motion planning is conducted on a planar 3R robot. (b) Robot begins moving without information on the target location. (c) Bear is placed at a random position, and the camera then determines its location. (d) SMMs belonging to this target location are computed immediately and the robot then begins to move toward the bounding box of the closest SMM. (e) Robot moves into the corresponding SMM bounding box and after 5.8 seconds, (f) joint 2 becomes locked. (g) Robot is still able to grasp the toy bear despite this joint failure.

Kinova Gen3 robot by only using three joints whose axes are parallel to each other, as shown in Fig. 17(a). Joint limits and self-collisions are not considered in these experiments. The goal of the experiment is to successfully grasp a toy bear, whose position is initially unknown, after the robot experiences an arbitrary locked-joint failure. The experiment begins with the robot moving without knowledge of the bear's location, as shown in Fig. 17(b). The bear is then placed at a random location in the robot's workspace, and an RGB-D camera system is used to detect the bear and obtain its location, as shown in Fig. 17(c). After the target workspace location is determined, the SMMs belonging to this workspace location are immediately computed using the proposed method. The bounding boxes of these SMMs are then computed and used to plan a fault-tolerant trajectory, where the robot is shown moving toward the SMM bounding box in Fig. 17(d) and is inside of the SMM bounding box in Fig. 17(e). Joint 2 then becomes locked after 5.75 s, as shown in Fig. 17(f). Despite this joint failure, the robot is still able to successfully grasp the bear, as shown in Fig. 17(g). A second experiment was performed with a failure of joint 3, and the robot was again able to successfully grasp the bear.[2]

[2] The physical experiment videos are available on YouTube at: https://youtu.be/orYHjaKYb2Y

VII. Conclusion

This work presents a learning-based method to accurately approximate the SMMs of arbitrary robots with a single degree of redundancy in an efficient manner for real-time global motion planning. To achieve this goal, a learnable closed-form representation of SMMs and a novel method of clustering workspace locations and grouping SMMs by homotopy classes are developed. Given an arbitrary workspace location, the associated set of homotopy classes is first predicted by a neural network, and the SMMs of these homotopy classes are then approximated by an additional neural network. On average, the proposed method computes the SMMs of workspace locations in 0.0014, 0.0016, and 0.0017 s for 3R, 4R, and 7R robots, respectively. This is much faster than the average runtimes of the nullspace projection method, the sampling-based method, and the grid-based method. This high computational efficiency is achieved while also retaining a high level of approximation accuracy with average FK errors of 0.29% , 0.45% and 2.09% , respectively. After applying a simple error correction mechanism, these average errors are reduced to 7.3×10^{-5}% , 0.0011% , and 0.016% , respectively. Furthermore, given the highly parallelizable characteristics of neural networks, the computational cost of the

proposed method does not scale directly with increases in the number of input workspace locations. As is illustrated by the fault-tolerant motion planning demonstration, this is the only method that can be used for real-time global motion planning based on the approximated SMMs.

This method can be further improved in the following aspects. First, the proposed method can be further extended to robots with multiple degrees of redundancy. This can be done by using redundant parameters to reduce the degrees of redundancy of a given robot to one and approximating the SMMs of this reduced robot with the proposed method. These approximated SMMs are slices of the high-dimensional SMMs along the redundant parameters. An alternative approach is to represent SMMs using the n-dimensional Fourier transform and learning these Fourier coefficients. Second, obstacle avoidance and joint limits will also be considered in future works to plan fault-tolerant collision-free trajectories. The SMM bounding boxes and collision-free polytopes will be intersected in the joint space and convex optimization algorithms can be used to compute optimal collision-free trajectories [31]. The above improvements will be considered in our future work.

REFERENCES

[1] I. Vitanov et al., "A suite of robotic solutions for nuclear waste decommissioning," *Robotics*, vol. 10, no. 4, 2021, Art. no. 112.

[2] Y. Gao and S. Chien, "Review on space robotics: Toward top-level science through space exploration," *Sci. Robot.*, vol. 2, no. 7, pp. 50–74, 2017.

[3] J. Delmerico et al., "The current state and future outlook of rescue robotics," *J. Field Robot.*, vol. 36, no. 7, pp. 1171–1191, 2019.

[4] Z. Vintr and M. Vintr, "Reliability prediction for components of robotic systems," in *Proc. Adv. Automat. Robot., Vol. 2: Sel. Papers 2011 Int. Conf. Automat. Robot.*, Dubai, 2012, pp. 463–470.

[5] B. Siciliano and O. Khatib, *Springer Handbook of Robotics*. Berlin, Germany: Springer, 2016.

[6] M. L. Visinsky, J. R. Cavallaro, and I. D. Walker, "Robotic fault detection and fault tolerance: A survey," *Rel. Eng. Syst. Saf.*, vol. 46, no. 2, pp. 139–158, 1994.

[7] B. Xie, J. Zhao, and Y. Liu, "Fault tolerant motion planning of robotic manipulators based on a nested RRT algorithm," *Ind. Robot: An Int. J.*, vol. 39, no. 1, pp. 40–46, 2012.

[8] C. L. Lewis and A. A. Maciejewski, "Fault tolerant operation of kinematically redundant manipulators for locked joint failures," *IEEE Trans. Robot. Automat.*, vol. 13, no. 4, pp. 622–629, Aug. 1997.

[9] B. Xie and A. A. Maciejewski, "Maximizing the probability of task completion for redundant robots experiencing locked joint failures," *IEEE Trans. Robot.*, vol. 38, no. 1, pp. 616–625, Feb. 2022.

[10] C. J. Paredis and P. K. Khosla, "Fault tolerant task execution through global trajectory planning," *Rel. Eng. Syst. Saf.*, vol. 53, no. 3, pp. 225–235, 1996.

[11] A. M. Bader and A. A. Maciejewski, "A hybrid approach for estimating the failure-tolerant workspace size of kinematically redundant robots," *IEEE Robot. Automat. Lett.*, vol. 6, no. 2, pp. 303–310, Apr. 2021.

[12] Z. Mu, L. Han, W. Xu, B. Li, and B. Liang, "Kinematic analysis and fault-tolerant trajectory planning of space manipulator under a single joint failure," *Robot. Biomimetics*, vol. 3, pp. 1–10, 2016.

[13] S. Wang, J. Z. Wen, D. Zhou, and Y. Wu, "A game-based fault-tolerant path planning algorithm for space manipulator," in *Proc. 2021 IEEE Int. Conf. Mechatron. Automat.*, 2021, pp. 871–877.

[14] K. Li and Y. Zhang, "Fault-tolerant motion planning and control of redundant manipulator," *Control Eng. Pract.*, vol. 20, no. 3, pp. 282–292, 2012.

[15] Z. Li, C. Li, S. Li, S. Zhu, and H. Samani, "A sparsity-based method for fault-tolerant manipulation of a redundant robot," *Robotica*, vol. 40, no. 10, pp. 3396–3414, 2022.

[16] R. G. Roberts and A. A. Maciejewski, "A local measure of fault tolerance for kinematically redundant manipulators," *IEEE Trans. Robot. Automat.*, vol. 12, no. 4, pp. 543–552, Aug. 1996.

[17] J. W. Burdick, "On the inverse kinematics of redundant manipulators: Characterization of the self-motion manifolds," in *Proc. Adv. Robot.: 4th Int. Conf. Adv. Robot.*, Columbus, OH, USA, Jun. 1989, pp. 25–34.

[18] A. A. Almarkhi and A. A. Maciejewski, "Maximizing the size of self-motion manifolds to improve robot fault tolerance," *IEEE Robot. Automat. Lett.*, vol. 4, no. 3, pp. 2653–2660, Jul. 2019.

[19] A. Peidro, O. Reinoso, A. Gil, J. M. Marín, and L. Paya, "A method based on the vanishing of self-motion manifolds to determine the collision-free workspace of redundant robots," *Mechanism Mach. Theory*, vol. 128, pp. 84–109, 2018.

[20] I. Banfield and H. Rodríguez, "Generation of the self-motion manifolds of a functionally redundant robot using multi-objective optimization," in *Robotics for Sustainable Future: CLAWAR 2021 24*. Berlin, Germany: Springer, 2022, pp. 438–452.

[21] M. Shimizu, H. Kakuya, W.-K. Yoon, K. Kitagaki, and K. Kosuge, "Analytical inverse kinematic computation for 7-DOF redundant manipulators with joint limits and its application to redundancy resolution," *IEEE Trans. Robot.*, vol. 24, no. 5, pp. 1131–1142, Oct. 2008.

[22] J. J. Rice and J. M. Schimmels, "Multi-homotopy class optimal path planning for manipulation with one degree of redundancy," *Mechanism Mach. Theory*, vol. 149, 2020, Art. no. 103834.

[23] T. Wu, J. Zhao, and B. Xie, "A novel method for computing self-motion manifolds," *Mechanism Mach. Theory*, vol. 179, pp. 105–121, 2023.

[24] D. E. DeMers, "Learning to invert many-to-one mappings," Ph.D. dissertation, Dept. Comput. Sci. Eng., Univ. California, San Diego, CA, USA, 1993.

[25] K. Tchoń and A. Matuszok, "On avoiding singularities in redundant robot kinematics," *Robotica*, vol. 13, no. 6, pp. 599–606, 1995.

[26] K. M. Ben-Gharbia, A. A. Maciejewski, and R. G. Roberts, "Kinematic design of redundant robotic manipulators for spatial positioning that are optimally fault tolerant," *IEEE Trans. Robot.*, vol. 29, no. 5, pp. 1300–1307, Oct. 2013.

[27] K. M. Ben-Gharbia, A. A. Maciejewski, and R. G. Roberts, "Kinematic design of manipulators with seven revolute joints optimized for fault tolerance," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 46, no. 10, pp. 1364–1373, Oct. 2016.

[28] S. Kim and J. Perez, "Learning reachable manifold and inverse mapping for a redundant robot manipulator," in *Proc. 2021 IEEE Int. Conf. Robot. Automat.*, 2021, pp. 4731–4737.

[29] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*.

[30] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Representations*, 2019.

[31] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Sci. Robot.*, vol. 8, no. 84, 2023, Art. no. eadf7843.

Charles L. Clark (Student Member, IEEE) received the B.Sc. degree in computer science from Xavier University, Cincinnati, OH, USA in 2020. He is currently working toward the Ph.D. degree in electrical engineering at the IRA Lab, University of Kentucky (U.K.), USA.

He is a Research Assistant in electrical engineering with the IRA Lab, U.K. His current research interests include machine learning, redundant robots, fault-tolerant robots, and robot motion planning.

Biyun Xie (Senior Member, IEEE) received the B.S. and the Ph.D. degrees in mechanical engineering from the Beijing University of Technology, Beijing, China, in 2009 and 2015, respectively, and the second Ph.D. degree in electrical engineering from Colorado State University, Fort Collins, CO, USA, in 2019.

She is currently an Assistant Professor of electrical and computer engineering with the University of Kentucky, Lexington, KY, USA. She has authored and coauthored more than 30 papers. Her research interests include redundant robots, fault-tolerant robots, and robot motion planning.

Dr. Xie is an Associate Editor for IROS, ICRA, and IEEE TRANSACTIONS ON ROBOTICS. She is also the Associate Vice President of the IEEE Robotics and Automation Society (RAS) Technical Activities Board and Co-Chair of the IEEE RAS TC on Algorithms for Planning and Control of Robot Motion.