Maximizing the Size of Self-Motion Manifolds to Improve Robot Fault Tolerance

Ahmad A. Almarkhi and Anthony A. Maciejewski

Abstract—**One measure of a robot's fault tolerance is the size of its self-motion manifold. This letter presents a new methodology for finding the largest self-motion manifold(s) of kinematically redundant robots, that consists of two algorithms. Because large self-motion manifolds occur near singular configurations, the first algorithm is designed to identify singularities of all ranks, including high-rank singularities. One unique feature of this algorithm is its ability to deal with the ill conditioned nature of singular vectors when there are multiple nearly equal singular values. The second algorithm is constructed to compute the self-motion manifolds that contain these singular configurations by iteratively moving along the null space of the robot's Jacobian. An important aspect of this computation is to deal with singularities along the manifold, where the null space is high dimensional. These two algorithms are applied to the well-known Mitsubishi PA-10 to illustrate their effectiveness at identifying singularities and computing the largest self-motion manifold(s).**

Index Terms—**Redundant Robots, Kinematics.**

I. Introduction

FAULT tolerance has been a critical factor in the design and operation of robotic systems that are meant to operate in harsh environments. Due to the mission-critical nature of some robotic applications, failure could result in catastrophic loss of life and/or property. Robots used in search and rescue operations launched after disasters are good examples of when reliability is important [1]. Previous work has shown that the availability of robots in such harsh environments is as low as 50% [2]. Because certain failures can put the entire mission in jeopardy [3], work has been done to redesign rescue robots to make them fault tolerant [4].

Many different aspects of fault tolerance have been considered, such as fault detection, identification, and analysis, as surveyed in [5]. Researchers have also looked at fault-tolerant control of actuators, for example, in automated underwater vehicles [6], [7]. Fault-tolerant control for multirobot systems with undetected failures is discussed in [8]. In all cases, fault tolerance requires redundancy at some level. Categories of redundancy include: *structural redundancy*, e.g., duplicating parts that are most susceptible to failures [9]; *functional redundancy*, i.e., a human operator intervenes to assess faults and implement a work-around; *analytical redundancy*, e.g., when a tachometer signal is integrated to recover from a failed position sensor [5]; and *kinematic redundancy*, i.e., a robot is designed to have more degrees of freedom (DoFs) than the minimum required to complete a task in order to compensate for the lost joint(s). The work presented here focuses on the study of kinematic redundancy.

To quantify the impact of incorporating kinematic redundancy, researchers have classified the measures of fault tolerance into two categories, i.e., local and global. Quantitative local fault tolerance measures are typically based on the singular value decomposition of the robot's Jacobian matrix. Such measures include the *minimum singular value* [10], the *condition number* [11], and the *robot manipulability* [12]. The kinematic redundancy is used to configure the robot so that it optimizes the fault tolerance measure. Techniques for doing so frequently involve the *gradient of the minimum singular value* [13].

Global fault-tolerance measures, that typically define reachable workspaces, are useful for pick-and-place tasks. One such measure [10] can be used to identify the best fault-tolerant location for these types of tasks. It quantifies the size of the workspace where the robot can operate before a failure and still return to the desired location after a failure. This can be guaranteed if the robot is operated within specific joint limits determined from the range of the robot's self-motion manifold. The problem becomes more challenging if the aim is to design a fault-tolerant workspace that is reachable for any trajectory both before and after a failure. A technique for computing the boundaries of such a workspace is presented in [14]. One can use these global measures to evaluate and select the optimal kinematic parameters when designing redundant robots with the same number of DoFs [15].

This work focuses on improving the global fault tolerance of a robot by maximizing the size of a pre-failure workspace while guaranteeing the reachability of a critical task location. This is done by identifying "large" self-motion manifolds, where the metric for the size depends on the ranges of each of the joints, i.e., their bounding box. It is shown that such large self-motion manifolds can be found by searching near high-rank singular configurations because these configurations represent connections of two or more previously disjoint manifolds.

The rest of this letter is organized as follows. An overview of the terminology and background concepts used is presented in Section II. In Section III, the approach used to analyze a robot design in order to identify its best fault tolerant location(s) is explained. This approach is then illustrated on a common existing

Manuscript received January 28, 2019; accepted April 23, 2019. Date of publication April 29, 2019; date of current version May 6, 2019. This letter was recommended for publication by Associate Editor J. Xiao and Editor D. Song upon evaluation of the reviewers' comments. *(Corresponding author: Ahmad A. Almarkhi.)*

The authors are with the Walter Scott, Jr. College of Engineering, Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523-1373 USA (e-mail: almarkhi@colostate.edu; aam@colostate.edu).

Digital Object Identifier 10.1109/LRA.2019.2913994

redundant robot design in Section IV. Finally, the conclusions of this work are presented in Section V.

II. Background

A. Self-Motion Manifolds

The forward kinematics of a robot is represented as

$$\boldsymbol{x} = f(\boldsymbol{\theta}) \tag{1}$$

where $\boldsymbol{x}$ is an m-dimensional vector representing the end-effector location (position and orientation) and θ is an n-dimensional vector representing the joint angles. For redundant robots, $n > m$, where $n - m$ is the degree of redundancy. In this case, the self-motion manifold(s) is (are) the set of all solutions that result from solving the inverse-kinematic problem represented by

$$\boldsymbol{\theta} = f^{-1}(\boldsymbol{x}). \tag{2}$$

The upper limit on the number of self-motion manifolds for redundant spherical, positional, and spatial manipulators is 2, 4, and 16, respectively [16]. The relationship between the robot's joint velocity and its end-effector velocity is represented by

$$\dot{\boldsymbol{x}} = \boldsymbol{J}\dot{\boldsymbol{\theta}} \tag{3}$$

where J is the $m \times n$ Jacobian. At the velocity level, self motion corresponds to:

$$\boldsymbol{J}\dot{\boldsymbol{\theta}} = \boldsymbol{0}. \tag{4}$$

For the case where $n - m = 1$ and the robot is in a non-singular configuration, the null space is one dimensional. In this case, the null space will be represented by the unit vector $\hat{\boldsymbol{n}}_J$, which is tangent to a self-motion manifold associated with this location. One can use $\hat{\boldsymbol{n}}_J$ to map out the self-motion manifold(s) by integrating how it evolves under the constraint of maintaining a fixed desired end-effector location, $\boldsymbol{x}_d$. Numerically, this can be done by identifying an initial configuration $\boldsymbol{\theta}_0$ where $\boldsymbol{x}_d = f(\boldsymbol{\theta}_0)$, and repeatedly solving

$$\Delta\boldsymbol{\theta} = \gamma\hat{\boldsymbol{n}}_J + \boldsymbol{J}^{+}\Delta\boldsymbol{x}_e \tag{5}$$

where $\Delta\boldsymbol{\theta}$ is the change in the joint angles, γ is a real positive scalar that represents the step size along the manifold, and $\boldsymbol{J}^{+}\Delta\boldsymbol{x}_e$ is an error correction term where $\boldsymbol{J}^{+}$ is the pseudoinverse of the Jacobian matrix and $\Delta\boldsymbol{x}_e$ is the end-effector error, i.e., the difference between $f(\boldsymbol{\theta} + \Delta\boldsymbol{\theta})$ and $\boldsymbol{x}_d$.[1] If there are multiple self-motion manifolds, this procedure must be performed on each of them with an appropriate initial $\boldsymbol{\theta}_0$. The characteristics of the individual manifolds can be significantly different in terms of their shape and size.

B. Size of Self-Motion Manifolds

To determine the length of a one-dimensional self-motion manifold, one only needs to sum up the number of times that (5) is solved to traverse the entire manifold. To identify when one has returned to the initial configuration, one must be careful to consider the case when one or more of the joints has rotated by 2π. More so than length, the range through which each joint angle moves is a useful measure of the robot's fault tolerance at the location associated with this manifold. These ranges define a bounding box, the volume of which has been used as one measure of fault tolerance [10]. Unfortunately, many common 7-DoF robot designs that are like a human arm, have self-motion manifolds that have a zero range for a particular joint, i.e., the elbow joint. Therefore, throughout this work we always use the sum of all joint angle ranges for all self-motion manifolds associated with a location as a measure of fault tolerance.

Fortunately, for higher degrees of redundancy (where $n - m > 1$) that result in higher dimensional self-motion manifolds, computing an estimate of a bounding box is more tractable than computing areas, volumes, or hypervolumes of manifolds. For these cases, a bounding box on the joint angle ranges can be computed by modifying (5) to

$$\Delta\boldsymbol{\theta} = \gamma\boldsymbol{N}_J\hat{\boldsymbol{e}}_i + \boldsymbol{J}^{+}\Delta\boldsymbol{x}_e \tag{6}$$

where $\boldsymbol{N}_J$ is a projection onto the $(n - m)$-dimensional null space of the Jacobian and $\hat{\boldsymbol{e}}_i$ is a unit vector along the ith joint angle, where $1 \leq i \leq n$ [10]. By repeatedly solving (6), for $i = 1$ to n, one can find an approximation of joint-angle ranges that can be used to compute the self-motion manifold size.[2] The iteration defined by (6) should be terminated when either joint angle i traverses 2π or the projection of $\hat{\boldsymbol{e}}_i$ onto the null space becomes zero. In the latter case, this may be a local minimum so that this measure is a lower bound on the range of joint i.

As illustrated in Fig. 1, singularities play a critical role in the size and shape of self-motion manifolds. At singularities, two (or more) self-motion manifolds can touch and become one manifold, or one manifold can tear apart. This means that larger manifolds tend to include one or more singularities and so it is natural to search for large manifolds near singularities.

III. Identifying Larger Self-Motion Manifolds

As discussed above, the larger (thus the more fault-tolerant) self-motion manifolds exist near singularities, so that one should employ a technique for identifying singular configurations. There are many techniques for doing so, e.g., symbolically solving for when the determinant of $\boldsymbol{J}$ becomes zero [17] or using reciprocity-based resolution [18]. However, here we employ a technique based on the gradients of the singular values [13] because of its ability to identify high-rank singularities. The singular value decomposition of $\boldsymbol{J}$ can be defined as

$$\boldsymbol{J} = \boldsymbol{U}\boldsymbol{D}\boldsymbol{V}^{\top} \tag{7}$$

where $\boldsymbol{U}$ is an $m \times m$ orthogonal matrix of the output singular vectors, $\boldsymbol{V}$ is an $n \times n$ orthogonal matrix of the input singular vectors, and $\boldsymbol{D}$ is an $m \times n$ diagonal matrix where its diagonal

[1]The case where θ results in a singular configuration and null space is multi-dimensional will be discussed in Section III.

[2]This letter will focus only on the 7-DoF case, but future work will expand to higher DoF robots.

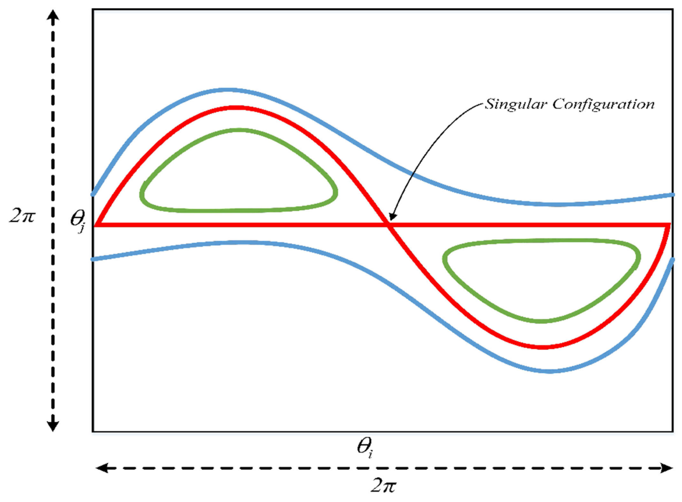

Fig. 1. An illustrative example of commonly occurring self-motion manifolds for high-dimensional manipulators is shown projected into the two-dimensional space for joints θ_i and θ_j. The single one-dimensional manifold shown in red is the connection of two previously disjoint manifolds. At the center of the figure, the intersection is a singular configuration where the null space is two-dimensional. Note that this is a true intersection and not simply due to the projection onto the θ_i - θ_j plane. If one perturbs the end-effector location from the one associated with the red manifold, the resulting manifolds can be quite different depending on the direction of the perturbation. In blue, the one red manifold splits into two open manifolds and in green into two closed manifolds, where open refers to the fact that θ_i can take on any value.

elements are the ordered singular values, i.e., $\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_m \geq 0$. It can be rewritten as a summation in terms of the singular vectors

$$\boldsymbol{J} = \sum_{i=1}^{r} \sigma_i \hat{\boldsymbol{u}}_i \hat{\boldsymbol{v}}_i^\top \tag{8}$$

where the vectors $\hat{\boldsymbol{u}}_i$ and $\hat{\boldsymbol{v}}_i$ represent the output and input singular vectors, respectively. For nonsingular $\boldsymbol{J}$, the value of σ_m represents the distance to a singularity. The rank of $\boldsymbol{J}$, denoted r, is less than m if the robot is singular (i.e, $\sigma_i = 0$ for $i > r$). In this case, the value of σ_r is the distance to the next higher-rank singularity. One can drive the robot towards the nearest singularity by moving the robot along the gradient of σ_m until it reaches zero. From (8), it is easy to see that any singular value σ_i can be written as

$$\sigma_i = \hat{\boldsymbol{u}}_i^\top \boldsymbol{J} \hat{\boldsymbol{v}}_i. \tag{9}$$

By differentiating (9) with respect to time, one obtains

$$\dot{\sigma_i} = \dot{\hat{\boldsymbol{u}}}_i^\top \boldsymbol{J} \hat{\boldsymbol{v}}_i + \hat{\boldsymbol{u}}_i^\top \dot{\boldsymbol{J}} \hat{\boldsymbol{v}}_i + \hat{\boldsymbol{u}}_i^\top \boldsymbol{J} \dot{\hat{\boldsymbol{v}}}_i \tag{10}$$

that can be simplified to

$$\dot{\sigma_i} = \hat{\boldsymbol{u}}_i^\top \dot{\boldsymbol{J}} \hat{\boldsymbol{v}}_i. \tag{11}$$

The partial derivative of σ_i with respect to θ_k can be written as

$$\frac{\partial \sigma_i}{\partial \theta_k} = \hat{\boldsymbol{u}}_i^\top \frac{\partial \boldsymbol{J}}{\partial \theta_k} \hat{\boldsymbol{v}}_i \tag{12}$$

where

$$\frac{\partial \boldsymbol{J}}{\partial \theta_k} = \left[\frac{\partial \boldsymbol{j}_1}{\partial \theta_k}, \frac{\partial \boldsymbol{j}_2}{\partial \theta_k}, \ldots, \frac{\partial \boldsymbol{j}_n}{\partial \theta_k}\right]. \tag{13}$$

The partial derivative of the i^{th} column of the Jacobian is given by [13], [19]

$$\frac{\partial \boldsymbol{j}_i}{\partial \theta_k} = \begin{cases} \begin{bmatrix} (\boldsymbol{z}_k^\top \boldsymbol{p}_i)\boldsymbol{z}_i - (\boldsymbol{z}_k^\top \boldsymbol{z}_i)\boldsymbol{p}_k \\ \boldsymbol{z}_k \times \boldsymbol{z}_i \end{bmatrix}, k < i \\ \begin{bmatrix} (\boldsymbol{z}_i^\top \boldsymbol{p}_k)\boldsymbol{z}_k - (\boldsymbol{z}_k^\top \boldsymbol{z}_i)\boldsymbol{p}_k \\ \boldsymbol{0} \end{bmatrix}, k \geq i \end{cases} \tag{14}$$

Utilizing (12), (13), and (14), one can easily compute the gradient of any singular value of $\boldsymbol{J}$ as

$$\nabla \sigma_i = \left[\frac{\partial \sigma_i}{\partial \theta_1}, \frac{\partial \sigma_i}{\partial \theta_2}, \ldots, \frac{\partial \sigma_i}{\partial \theta_n}\right]. \tag{15}$$

Moving along this gradient allows one to increase or decrease any desired singular value.

Finding the largest self-motion manifold(s) of any robot is a two-step process. The first step is to find all singular configurations, including high-rank singularities. In this step, we employ the gradient descent technique to minimize a desired singular value

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha \nabla \sigma_i \tag{16}$$

where $\boldsymbol{\theta}^{(k)}$ is the current joint configuration, $\boldsymbol{\theta}^{(k+1)}$ \unboldmath is the next joint configuration, and α is a positive scalar, often referred to as the step size. To identify rank-1 singularities, we start with generating random configurations in the joint space. Then, starting from each random configuration, we move the robot along the negative direction of the gradient of σ_m until the value of σ_m approaches zero. The set of final configurations that satisfy the condition of $\sigma_m \leq \epsilon$ are the rank-1 singularities of the robot, where ϵ is a user-defined threshold.

To identify the rank-2 singularities, we start from the same random configurations used to identify rank-1 singularities, but the robot is first moved along the negative direction of $\nabla\sigma_{m-1}$ until the value of σ_{m-1} approaches zero, i.e, $\sigma_m \leq \sigma_{m-1} \leq \epsilon$. In some cases σ_{m-1} and σ_m become nearly equal before σ_{m-1} approaches zero. This means that the singular vectors $\hat{\boldsymbol{u}}_{m-1}$ and $\hat{\boldsymbol{u}}_m$ (as well as $\hat{\boldsymbol{v}}_{m-1}$ and $\hat{\boldsymbol{v}}_m$) are ill-defined. That is, any vectors in the subspaces $\{\hat{\boldsymbol{u}}_{m-1}, \hat{\boldsymbol{u}}_m\}$ and $\{\hat{\boldsymbol{v}}_{m-1}, \hat{\boldsymbol{v}}_m\}$ are valid singular vectors for the gradient computation in (11) and (12). In such cases, we rotate the $\{\hat{\boldsymbol{u}}_{m-1}, \hat{\boldsymbol{u}}_m\}$ and $\{\hat{\boldsymbol{v}}_{m-1}, \hat{\boldsymbol{v}}_m\}$ subspaces so that the angle between $\nabla\sigma_m$ and $\nabla\sigma_{m-1}$ is minimized. (The angle between the gradients of σ_m and σ_{m-1} may vary from 0 to π.) We then reduce σ_{m-1} by moving along a negative direction of a linear combination of the gradients of σ_m and σ_{m-1}. We optimize the linear combination by doing two one-dimensional searches. The first search is along $\beta\nabla\sigma_m + (1-\beta)\nabla\sigma_{m-1}$ to determine the optimal value of β, where $0 \leq \beta \leq 1$, that minimizes σ_{m-1}. The second, is to determine the optimal value of the adaptive step size α along the negative direction of the computed combination. Those configurations that converge to where $\sigma_m \leq \sigma_{m-1} \leq \epsilon$ are rank-2 singularities. If the process does not converge, then a rank-2 singularity does not exist near this configuration. An analogous procedure is used to find rank-3 (and higher-rank) singularities. There are two cases where the singular vectors are ill-defined, i.e., when σ_{m-2} and σ_{m-1} are nearly equal or if the three singular values σ_{m-2}, σ_{m-1}, and σ_m are all

Algorithm 1: Identify All-rank Singular Configurations.

```
1:  select N random joint-space configurations[3]
2:  for $i = 6$ to 1 do {for each workspace dimension}
3:    for $j = 1$ to $N$ do {for each random configuration}
4:      select $j^{th}$ joint-space configuration $\boldsymbol{\theta}$
5:      compute $\sigma_i$ {robot Jacobian's $i^{th}$ singular value}
6:      while $\sigma_i \geq \epsilon$ do
7:        if $(i = 6)$ then {for rank-one singularities}
8:          $\nabla\sigma = \nabla\sigma_6$
9:        else {for high-rank singularities}
10:         for all $\sigma_k$ where $\sigma_i \approx \sigma_k$ do {where $k < i$}
11:           rotate $\boldsymbol{U}$ and $\boldsymbol{V}$ subspace associated with $\sigma_i$
              and $\sigma_k$'s {to minimize the angles between
              $\nabla\sigma_i, \nabla\sigma_{i+1}, \ldots, \nabla\sigma_k$}
12:           compute $\nabla\sigma$ {optimal linear combination of
              the gradients of the singular values}
13:         end for
14:       end if
15:       compute $\alpha$ {adaptive linear search along $\nabla\sigma$}
16:       update $\boldsymbol{\theta}$ {$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} - \alpha\nabla\sigma$}
17:       if ($\sigma_i$ did not decrease) then
18:         go to 23 {local minimum of $\sigma_i$}
19:       end if
20:     end while
21:     save $\boldsymbol{\theta}$ and singularity rank
22:   end for
23: end for
```

nearly equal. In the first case, the optimization described above can be performed for σ_{m-2} and σ_{m-1} to minimize σ_{m-2}. In the second case, one needs to search for a suitable rotation for the $\{\hat{\boldsymbol{u}}_{m-2}, \hat{\boldsymbol{u}}_{m-1}, \hat{\boldsymbol{u}}_m\}$ and $\{\hat{\boldsymbol{v}}_{m-2}, \hat{\boldsymbol{v}}_{m-1}, \hat{\boldsymbol{v}}_m\}$ subspaces to minimize the sum of the angles between the gradients of the three singular values. Then, one must find a suitable combination of the gradients and a step size that minimize σ_{m-2}. An analogous process is repeated for higher-rank singularities until there is no possible joint motion that will reduce σ_r while keeping $\sigma_m \leq \sigma_{m-1} \leq \cdots \leq \sigma_{r+1} \leq \epsilon$, i.e., there are no rank-$(m-r)$ singularities. The pseudocode for performing this procedure is given in Algorithm 1.

The second step is to compute all the self-motion manifolds that include these singular configurations. However, one would like to reduce the number of these configurations, in order to reduce the amount of computations that result in very similar manifolds. Therefore, if two singular configurations are close to each other, then only one of them will have its self-motion manifold computed. Finding the self-motion manifolds for each singularity configuration can be done by starting the robot in that singular configuration and then repeatedly solving (5) until an entire self-motion manifold is computed.

Before applying (5), one must first compute the end-effector location associated with this singular configuration. Then, from

[3]The value of N should be large enough to sufficiently span the joint space. An analysis of this user-defined parameter is provided in Subsection IV-C below.

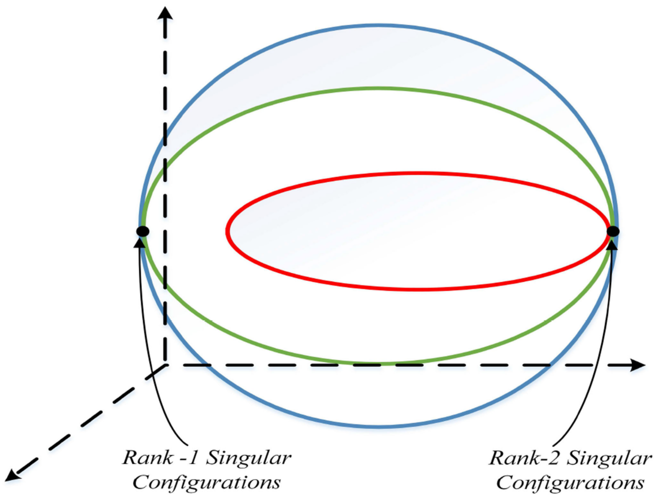
Rank -1 Singular Configurations
Rank-2 Singular Configurations

Fig. 2. This sketch illustrates the case where three previously disjointed manifolds (red, green, and blue) touch and become one manifold, i.e., a rank-2 singularity. Also, two of these manifolds (green and blue) touch elsewhere creating a rank-1 singularity. Many variations of different-rank singularities can exist on a single manifold.

the singular configuration, the robot is moved in each of the $(n-r)$ directions of the null space. Mathematically,

$$\Delta\boldsymbol{\theta} = \gamma\hat{\boldsymbol{v}}_i + \boldsymbol{J}^{+}\Delta\boldsymbol{x}_e \tag{17}$$

for $i = r$ to n, where these $\hat{\boldsymbol{v}}_i$ are the singular vectors that span the $(n-r)$-dimensional null space of the Jacobian at the singular configuration. This guarantees that all of the one-dimensional self-motion manifolds that touch at this configuration can be computed (see Fig. 1), because away from the singular configuration the null space becomes one dimensional and well defined. Therefore one can repeatedly solve (5) until the robot returns to the initial starting configuration.

However, it is common for self-motion manifolds to include multiple singular configurations (see Fig. 2). If while solving (5) the null space becomes multi-dimensional, i.e., another singularity is encountered, then one should be careful to select a null-space vector from this higher dimensional subspace that is as close as possible to the one used to enter the singularity. For computational efficiency, one should check any new starting configuration with all previously computed manifolds to prevent redundant computations. The pseudocode for performing the second step is given in Algorithm 2.

In general, an end-effector location will have multiple disjoint self-motion manifolds, i.e., the robot cannot move from one manifold to another without changing the end-effector location. To compute the "fault tolerance" of the end-effector location associated with a singular configuration, one needs to compute the "bounding box" of all the self-motion manifolds, i.e., the ranges for each of the joints while staying at this end-effector location. Because of multiple disjoint self-motion manifolds one needs to decide if it is important to reconfigure the arm from one manifold to another without moving the end effector. If it is not, then one can simply take the union of all the joint-angle ranges for each manifold. If it is, one can only take the union for those

Algorithm 2: Compute all Self-motion Manifolds with Singularities.

```
1:  import N singular configurations and their ranks {from algorithm} 1
    {remove duplicate singularities}
2:  $\forall i, j \le N, i \ne j$
3:  if $\boldsymbol{\theta}(i) \approx \boldsymbol{\theta}(j)$ then
4:    delete $\boldsymbol{\theta}(j)$
5:  end if
    {compute all self-motion manifolds SMMs}
6:  for each singularity rank do
7:    for all singular configurations $\boldsymbol{\theta}$ do {of each rank}
8:      if $\boldsymbol{\theta}$ does not exist on a previously computed manifold then
9:        compute $\boldsymbol{x}_e$ {the end-effector location}
10:       find the $n-r$ configurations near $\boldsymbol{\theta}$ that satisfy the $n-r$ singular directions at $\boldsymbol{x}_e$
11:       for $k = 1$ to $n-r$ do
12:         if $k^{th}$ configuration does not exists on a computed manifold then
13:           start at the $k^{th}$ configuration
14:           while not back to starting configuration do
15:             compute the robot Jacobian ($\boldsymbol{J}$)
16:             compute the null vector ($\hat{\boldsymbol{n}}_J$)
17:             compute $\Delta\boldsymbol{\theta}$ {$\Delta\boldsymbol{\theta} = \gamma\hat{\boldsymbol{n}}_J + \boldsymbol{J}^{+}\Delta\boldsymbol{x}_e$}
18:             update joint angles {$\boldsymbol{\theta}_{new} = \boldsymbol{\theta}_{old} + \Delta\boldsymbol{\theta}$}
19:           end while
20:         end if
21:       end for
22:     end if
23:   end for
24: end for
```

manifolds that touch. One measure of the size of the bounding box is simply the summation of all these joint-angle ranges. The above procedure for identifying the largest self-motion manifold is illustrated for the well-known 7-DoF Mitsubishi PA-10 robot in the next section.

IV. PA-10 Robot Case Study

A. PA-10 Background

The Mitsubishi PA-10 is used as an illustrative example because it has a well-known, commonly occurring 7-DoF kinematic structure. We first explain the behavior of the self-motion manifolds of the PA-10 when the end-effector location is changed. Then, the algorithms described in the previous section are used to identify the largest self-motion manifolds and how they relate to fault tolerance. The original DH parameters for the PA-10 are given in Table I [20], with the end effector positioned so that the last link's displacement, d_7, is equal to d_3.

Because the elbow joint, i.e., θ_4, is the only one that can change the distance between the shoulder and the wrist, there can be no component of θ_4 during self motion. Therefore, it is possible to categorize self-motion manifolds based on the value of θ_4. This is illustrated in Fig. 3 where a single manifold for each end-effector location is shown with θ_4 ranging from $-\pi$ to π. All of the singularities shown in this figure are rank-1 singularities.

TABLE I
THE DH PARAMETERS OF THE PA-10 ROBOT

$Link_i$	$\alpha_i[rad]$	$a_i[m]$	$d_i[m]$	$\theta_i[rad]$
1	$-\pi/2$	0	0	θ_1
2	$\pi/2$	0	0	θ_2
3	$-\pi/2$	0	0.45	θ_3
4	$\pi/2$	0	0	θ_4
5	$-\pi/2$	0	0.50	θ_5
6	$\pi/2$	0	0	θ_6
7	0	0	0.45	θ_7

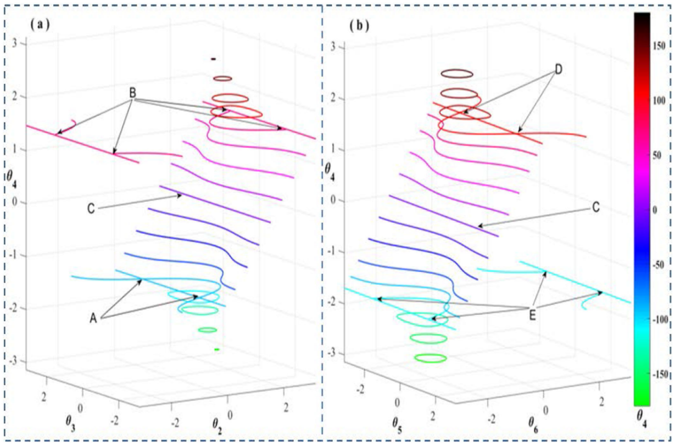

Fig. 3. This figure shows 3-D projections of the PA-10 self-motion manifolds generated by changing θ_4 from $-\pi$ to $+\pi$, where the value of θ_4 is indicated using color. Subfigure (a) is a projection in $[\theta_2, \theta_3, \theta_4]$ and (b) is a projection in $[\theta_6, \theta_5, \theta_4]$. Singularities occur at (A) where $\theta_2 = 0, \theta_3 = \pm\pi/2$, (B) where $\theta_2 = \pm\pi, \theta_3 = \pm\pi/2$, (C) where $\theta_4 = 0$, (D) where $\theta_6 = 0, \theta_5 = \pm\pi/2$, and (E) where $\theta_6 = \pm\pi, \theta_3 = \pm\pi/2$.

The self-motion manifolds in Fig. 3 exhibit all of the properties shown in Fig. 1, i.e., there are both open and closed manifolds with manifolds connecting/separating at singularities. It is also clear that the largest manifolds are those that include singularities.

The singularities of the PA-10 have been well studied [18], [21]. However, it is important to note that Algorithm 1 above can be applied to any arbitrary robot structure and its computational complexity does not change for high-rank singularities, which are important for identifying the largest self-motion manifold.

B. The Largest Self-Motion Manifold

Once all the singularities are identified, Algorithm 2 is able to compute the size of self-motion manifolds that include these singularities. It identified the largest self-motion manifold to be 35.90 rad,[4] where the ranges of $\theta_1, \theta_2, \theta_3, \theta_5,$ and θ_7 are

[4]This is the theoretical maximum size where the actual value would include any joint limits on θ_2 and θ_6 that are due to self collision.

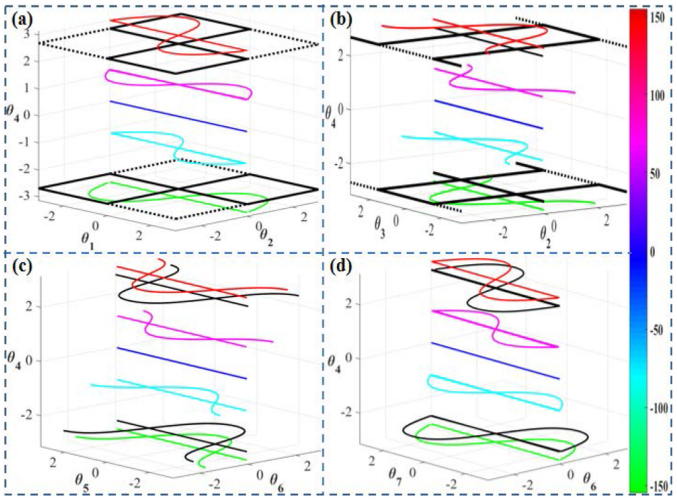

Fig. 4. The subfigures (a)-(d) show different 3-D projections of the PA-10 self-motion manifolds characterized by θ_4, where any color corresponds to only a single manifold. The optimal self-motion manifolds identified by Algorithm 2 are shown in black where $\theta_4 = \pm 2.69$ rad. The dotted lines are used to better show parts of the manifold that did not exist on nearby manifolds. The red and green manifolds where $\theta_4 = 3.00$ rad and $\theta_4 = -3.00$ rad, respectively, are shown to illustrate the behavior at a singularity.

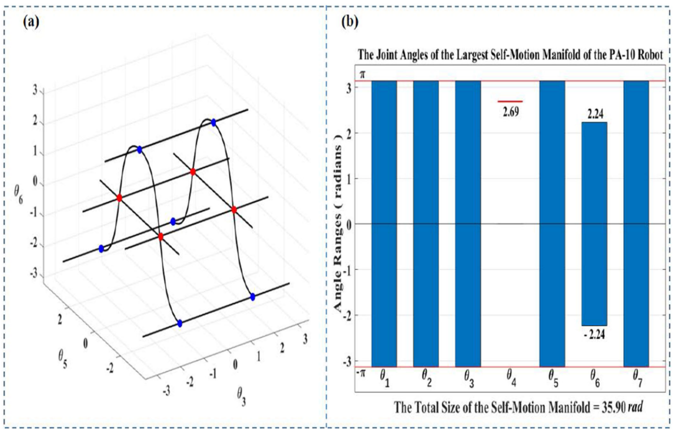

Fig. 5. In (a) the largest optimal self-motion manifold is shown projected into the θ_3, θ_5, and θ_6 subspace. From this projection one can see that this manifold contains four rank-1 and four rank-2 singularities, shown in blue and red respectively. Note that the two blue singularities at the bottom of the figure, i.e., where $\theta_6 = -2.24$ rad, are shown twice at both $\theta_5 = \pm\pi$. The rank-2 singularities occur when $\boldsymbol{\theta} = [0, 0, \frac{\pm\pi}{2}, \theta_4, \frac{\pm\pi}{2}, 0, 0]$, where in this case $\theta_4 = 2.69$ rad. The ranges of each of the joints is shown in (b) where $\theta_1, \theta_2, \theta_3, \theta_5$, and θ_7 are 2π, the range of θ_6 is 4.48 rad (± 2.24), and the value of θ_4 is 2.69 rad, where its range is zero.

2π rad, the range of θ_6 is 4.48 rad (± 2.24 rad),[5] and the range of θ_4 is zero, where its value is either $+2.69$ or -2.69 rad. To help understand why Algorithm 2 identified this as the optimal solution, consider Fig. 4 that shows several self-motion manifolds and their associated singularities characterized by varying θ_4 from $-\pi$ to π, analogous to Fig. 3. The black manifolds, where $\theta_4 = \pm 2.69$ rad, are clearly the largest self-motion manifolds. The nearby manifolds shown in green and red, where $\theta_4 = -3.00$ rad and $\theta_4 = 3.00$ rad, respectively, are shown to illustrate the behavior at these singularities. The subfigures (a)–(d) are all different projections of the same manifolds, i.e., any color corresponds to only a single manifold. These projections have been selected to illustrate which joint angles have an unrestricted range, i.e., $\theta_1, \theta_2, \theta_3, \theta_5$, and θ_7, whereas θ_6 has a range of ± 2.24 rad, which is clearly shown in (c) and (d).

One should note that the two black manifolds correspond to the same end-effector location, however, the robot cannot move from one manifold to the other without changing this location. The rank of the various singularities on the black manifold is not clear from Fig. 4 because they all appear to be of rank 1 due to the projections used. If one looks at the projection in θ_3, θ_5, and θ_6 space, as in Fig. 5(a), then it becomes clear that the black manifold contains four rank-1 and four rank-2 singularities, shown in blue and red respectively. (The four blue dots, rank-1 singularities, at the lower part of Fig. 5(a) only represent two singularities, i.e., the dots at $\theta_5 = \pi$ are the same as those at $\theta_5 = -\pi$).

Fig. 5(b) shows the ranges of the seven joint angles of the PA-10 while operating on the largest self-motion manifold. There are two separate, equal-sized manifolds at this location. The other manifold has identical joint-angle ranges, except that $\theta_4 = -2.69$ rad. The configurations with $\theta_4 = \pm 2.69$ rad are special because they make the axis between the shoulder and the wrist horizontal, and the rotation around this axis can configure the robot into four rank-1 singularities and four rank-2 singularities. Operating the robot slightly away from these special values of θ_4 will not allow it to reach all the rank-1 singularities. This will not dramatically change the size of the self-motion manifold, however the robot will lose some ability to reconfigure that is offered by the rank-1 singularities.

TABLE II
EFFICIENCY COMPARISON BETWEEN THE PROPOSED ALGORITHMS AND THE RANDOM APPROACH

No. of Random Configurations	Size of the Largest Self-Motion Manifold (Radians)	
	Random Approach	Proposed Algorithms
10	27.89	28.36
20	33.78	34.09
100	34.33	34.43
200	33.95	34.78
1,000	34.82	35.80
2,000	35.09	35.90
10,000	35.52	35.90

C. Evaluation

We first compare the proposed approach for identifying the largest self-motion manifold with a straightforward evaluation of self-motion manifolds generated at random configurations. Table II shows a comparison of the largest self-motion manifold identified by both techniques as a function of the number of random configurations N.

The data shows that the proposed approach converges to the maximum self-motion manifold size of 35.90 rad at $N = 2000$.

[5] If one wanted to modify the link offsets of the PA-10 to make $d_3 = d_5$, then the size of the largest self-motion manifold could be increased so that θ_6 would also have a range of 2π. This would also change the optimal value of θ_4 to 0.

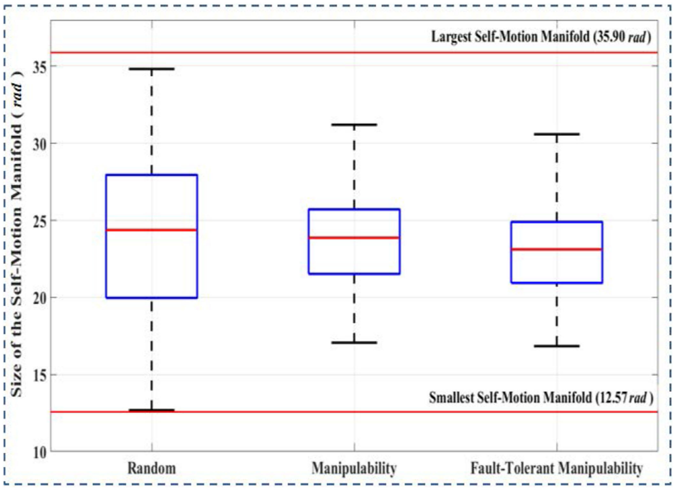

Fig. 6. This figure shows a comparison of the distributions of self-motion manifold sizes computed for 1,000 random configurations (left) with those found for 1,000 configurations with the best local dexterity measure (middle) and fault-tolerant measure (right). The best 1000 manipulability and fault-tolerant manipulability configurations were selected from 10,000 random configurations. The minimum and the maximum sizes in each distribution are indicated with black horizontal lines and the red line is the median. The largest (35.90 rad) and smallest (12.57 rad) self-motion manifold sizes for this robot are also indicated.

This was verified using ten different populations of random configurations. It should also be noted that our proposed approach outperformed the random approach for any value of N.

Next, we compared our proposed algorithm to an approach that attempts to identify large self-motion manifolds using classical local dexterity [22] and fault-tolerance measures [12].

Fig. 6 shows the distribution of $1,000$ self-motion manifold sizes computed from joint space configurations selected in three different ways. The left distribution is from the $N = 1,000$ entry in Table II, which is generated randomly. The middle and the right distributions are for the $1,000$ configurations with the best manipulability [22] and fault-tolerant manipulability [12], respectively, selected from $10,000$ random configurations. Note that the distribution of the self-motion manifold sizes generated from traditional local measures of dexterity and fault tolerance are outperformed by the random approach. This indicates that there is no correlation between classical local measures and self-motion manifold size. In addition, none of these techniques is able to identify the largest self-motion manifold of the robot. However, it is possible to use our proposed approach to optimize both global fault tolerance, i.e., largest self-motion manifold size, and any desired local measure of dexterity or fault tolerance. For example, Fig. 7(d) shows the PA-10 robot in one configuration on the largest self-motion manifold that minimizes the condition number of the Jacobian, i.e., $\sigma_1/\sigma_6 = 13.86$. This illustrates that one does not have to operate near a singular configuration in order to obtain the benefits of a fault-tolerant location with a large self-motion manifold.

D. Example Use Case

We now present a simple example use case where the performance of the proposed technique is compared to existing approaches [12], [22]. Assume that a robot will be employed in a remote environment where repair is not feasible and one is designing the workspace to determine where a critical task should be placed. The home configuration of the robot is where all tasks start, and one would like to be able to reach the goal location even after any single locked-joint failure.

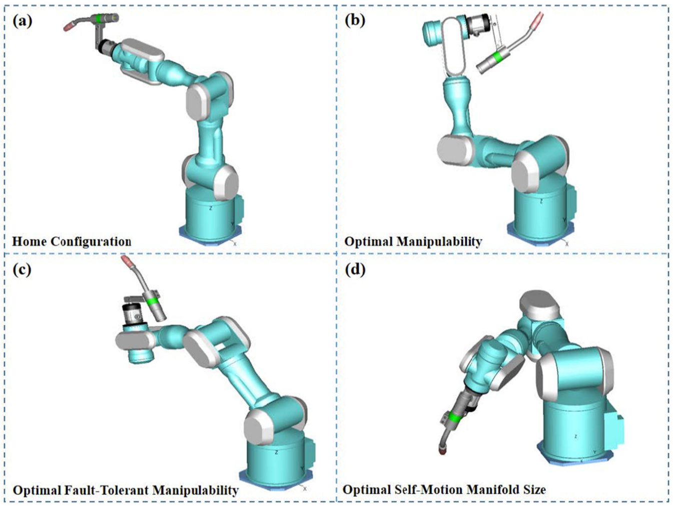
(a)
Home Configuration
(b)
Optimal Manipulability
(c)
Optimal Fault-Tolerant Manipulability
(d)
Optimal Self-Motion Manifold Size

Fig. 7. The PA-10 robot (with an arc-welding tool attached) is shown operating at multiple locations.[6] Subfigure (a) shows it in the home location, (b) in the best robot manipulability, (c) in the best fault-tolerant manipulability, and (d) in a location with the largest self-motion manifold. The self-motion manifold size associated with the best manipulability configuration is 23.67 rad, and for the fault-tolerant manipulability is 22.45 rad.

TABLE III
ROBOT'S HOME AND TASK CONFIGURATIONS

Start and Optimal Goals	Joint Configurations [rad]
Task Starting Location	$[0, 0, \frac{\pi}{2}, \frac{\pi}{2}, \frac{\pi}{2}, 0, 0]$
Manipulability	$[0, -1.8, -2.8, -1.3, -2.3, 1.5, 0]$
Fault-Tolerant Manipulability	$[0, -0.8, 3.0, 1.2, -2.9, 1.6, 0]$
Largest Self-motion Manifold	$[0, -\frac{\pi}{2}, \frac{\pi}{2}, -2.69, 1.14, \frac{\pi}{2}, 0]$

The start location is shown in Fig. 7(a). We select the goal location using the technique presented here, i.e., the location with the largest self-motion manifold, Fig. 7(d), and compare it to goal locations with globally optimal local measures, i.e., manipulability, Fig. 7(b), and fault-tolerant manipulability, Fig. 7(c). The joint values for all these configurations are given in Table III.

We then simulate a joint failure in each of the joints and perform inverse kinematics on the remaining six working joints and attempt to reach the goal locations. In all cases, the robot is able to reach location Fig. 7(d), however, there are several joint failures that prevent the robot from reaching the goal locations shown in Fig. 7(b) and (c). One example is shown in Fig. 8, where the robot is not able to reach the desired goal location due to a failure of joint six at $\theta_6 = 0$ that occurred at the start location. In both cases, this joint failure results in the desired goal location being outside the workspace of the damaged robots. Therefore, the best they can do is get to the closest location that is at the boundary of their new workspace. In Fig. 8 we select the "closest" configuration by minimizing the orientation error, so that all error is in the position of the tool. This illustrates the

[6]Fig. 7 and Fig. 8 were generated by using the Workspace 5 software package from WAT Solutions, (www.watsolutions.com).

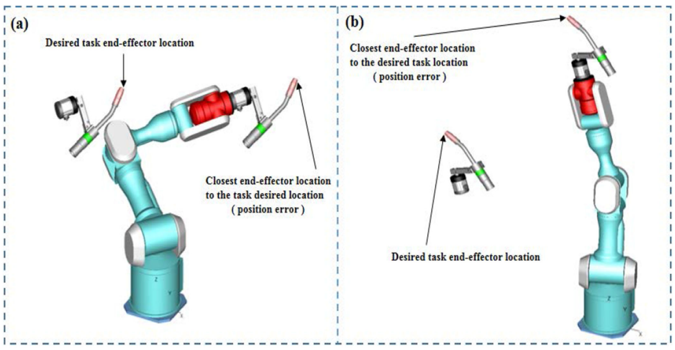
(a)
Desired task end-effector location
Closest end-effector location
to the task desired location
(position error)
(b)
Closest end-effector location
to the desired task location
(position error)
Desired task end-effector location

Fig. 8. The PA-10 robot is shown with joint six (in red) failed at $\theta_6 = 0$ while trying to perform a task where the goal location is either optimal manipulability in (a) or fault-tolerant manipulability in (b). An arc-welding tool is shown in the desired location to show the difference in position from the closest possible location for the damaged robots. The position errors in (a) and (b) are $[-0.66, -0.44, 0.02]m$ and $[-0.56, -0.14, -0.79]m$, respectively.

merit of operating a robot on the largest self-motion manifold for mission-critical tasks.

V. Conclusions

This work uses a measure of fault tolerance that is based on the size of self-motion manifolds. Because singularities occur at the connection of self-motion manifolds, they can be used to identify where larger manifolds exist. We developed algorithms that use this fact to; (1) first identify all ranks of singularities and then, (2) search in the proximity of these singularities to identify large self-motion manifolds. A unique feature of Algorithm 1 is that it can efficiently identify high-rank singularities for arbitrary robot structures. To do this it must track multiple singular values that are nearly equal, where their gradients are not well defined. Algorithm 2 also must deal with the ill-conditioned nature of singular vectors that occur at singular configurations. The efficacy of these algorithms is illustrated on a commonly occurring 7 DoF kinematic structure (Mistubishi PA-10). In addition to identifying the largest self-motion manifold, it provided information that allows one to modify the kinematics to obtain an even larger manifold. It was also able to identify joints that are fault intolerant, so that one could explore alternate designs.

References

[1] F. Matsuno and S. Tadokoro, "Rescue robots and systems in Japan," in *Proc. IEEE Int. Conf. Robot. Biomimetics*, 2004, pp. 12–20.

[2] J. Carlson and R. R. Murphy, "How UGVs physically fail in the field," *IEEE Trans. Robot.*, vol. 21, no. 3, pp. 423–437, Jun. 2005.

[3] K. Nagatani *et al.*, "Emergency response to the nuclear accident at the Fukushima Daiichi nuclear power plants using mobile rescue robots," *J. Field Robot.*, vol. 30, no. 1, pp. 44–63, 2013.

[4] K. Nagatani *et al.*, "Redesign of rescue mobile robot Quince," in *Proc. IEEE Int. Symp. Saf. Secur. Rescue Robot.*, 2011, pp. 13–18.

[5] 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.

[6] T. K. Podder, G. Antonelli, and N. Sarkar, "Fault tolerant control of an autonomous underwater vehicle under thruster redundancy: Simulations and experiments," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2000, pp. 1251–1256.

[7] N. Ranganathan, M. I. Patel, and R. Sathyamurthy, "An intelligent system for failure detection and control in an autonomous underwater vehicle," *IEEE Trans. Syst., Man, Cybern. – Part A: Syst. Humans*, vol. 31, no. 6, pp. 762–767, Nov. 2001.

[8] H. Park and S. A. Hutchinson, "Fault-tolerant rendezvous of multirobot systems," *IEEE Trans. Robot.*, vol. 33, no. 3, pp. 565–582, Jun. 2017.

[9] V. Monteverde and S. Tosunoglu, "Effect of kinematic structure and dual actuation on fault tolerance of robot manipulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, 1997, pp. 2902–2907.

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

[11] H. Abdi and S. Nahavandi, "Minimum reconfiguration for fault tolerant manipulators," in *Proc. 34th Annu. Mech. Robot. Conf., Parts A and B*, 2010, pp. 1345–1350.

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

[13] K. N. Groom, A. A. Maciejewski, and V. Balakrishnan, "Real-time failure-tolerant control of kinematically redundant manipulators," *IEEE Trans. Robot. Autom.*, vol. 15, no. 6, pp. 1109–1115, Dec. 1999.

[14] R. C. Hoover, R. G. Roberts, A. A. Maciejewski, P. S. Naik, and K. M. Ben-Gharbia, "Designing a failure-tolerant workspace for kinematically redundant robots," *IEEE Trans. Autom. Sci. Eng.*, vol. 12, no. 4, pp. 1421–1432, Oct. 2015.

[15] K. M. Ben-Gharbia, A. A. Maciejewski, and R. G. Roberts, "Modifying the kinematic structure of an anthropomorphic arm to improve fault tolerance," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2015, pp. 1455–1460.

[16] J. W. Burdick, *Kinematic Analysis and Design of Redundant Robot Manipulators*. Ph.D. dissertation, Dept. Comput. Sci., Stanford Univ., Stanford, CA, USA, 1988.

[17] K. Waldron, S.-L. Wang, and S. Bolin, "A study of the Jacobian matrix of serial manipulators," *J. Mech., Transmiss., Autom. Des.*, vol. 107, no. 2, pp. 230–237, 1985.

[18] S. B. Nokleby and R. P. Podhorodeski, "Reciprocity-based resolution of velocity degeneracies (singularities) for redundant manipulators," *Mech. Mach. Theory*, vol. 36, no. 3, pp. 397–409, 2001.

[19] C. A. Klein and L.-C. Chu, "Comparison of extended Jacobian and Lagrange multiplier based methods for resolving kinematic redundancy," *J. Intell. Robot. Syst.*, vol. 19, no. 1, pp. 39–54, 1997.

[20] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *ASME J. Appl. Mech.*, vol. 22, no. 4, pp. 215–221, 1955.

[21] S. B. Nokleby and R. P. Podhorodeski, "Identifying multi-DOF-loss velocity degeneracies in kinematically-redundant manipulators," *Mech. Mach. Theory*, vol. 39, no. 2, pp. 201–213, 2004.

[22] T. Yoshikawa, "Manipulability of robotic mechanisms," *Int. J. Robot. Res.*, vol. 4, no. 2, pp. 3–9, 1985.