2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Optimal Dexterity Path Planning for Robotic Manipulators Using Rapid Workspace Density Approximation

Nate S. Osikowicz[1], John R. Cooper[2], and Puneet Singla[3]

***Abstract*— This paper introduces a path planning algorithm for executing robotic manipulation tasks with maximum dexterity in the workspace. This is achieved by using the workspace density of the end-effector as the objective function in a sampling-based planner. In doing so, the path planning algorithm prioritizes joint configurations that correspond to the highest density of local end-effector positions. This results in a singularity avoidant path planning algorithm that favors redundancy, making it a favorable approach for manipulation scenarios in which dexterity is paramount. However, due to the exponential relationship between the number of possible end-effector positions and the number of joints, computing the workspace density via traditional methods is computationally intractable for most modern industrial robots. In this paper, a newly developed approach is taken wherein the workspace density is approximated by a Gaussian mixture model that solves for the optimal workspace density function subject to higher-order statistical moment constraints. The statistical moments of the workspace density function are computed recursively with a minimum number of sample points by using a non-product quadrature rule known as the Conjugate Unscented Transform (CUT). This results in a computationally efficient framework that allows the user to trade accuracy and computation time by varying the number of mixture components and the number of statistical moments used in the workspace density approximation. To demonstrate, the algorithm is implemented on the Precision Assembled Space Structure (PASS) platform at NASA illustrating its effectiveness in dexterous robotic assembly tasks.**

I. INTRODUCTION

In this paper, a novel workspace density approximation is combined with an optimal sampling-based path planner to generate robotic manipulation paths that maximize dexterity. Having the ability to maximize the dexterity of the end-effector is a desirable capability in robotic manipulation. The definition of dexterity varies slightly throughout the robotics literature, but generally refers to a robot's ability to move and apply forces in arbitrary directions as easily as possible [1], [2]. A distinction is made between the local dexterity, which measures the end-effector's motion capability in a particular pose, and the global dexterity, which measures the robot's dexterity across the entire workspace. To evaluate local dexterity, approaches in the literature generally seek a metric that quantifies the end-effector's distance from singularity. Yoshikawa introduced the manipulability measure $\mu = \sqrt{det(JJ^T)}$ by using the robot's Jacobian, J, to establish a link between dexterity and volume of the manipulability ellipsoid [3]. Another approach introduced by Salisbury and Craig [4] uses the Jacobian's condition number as a measure of proximity to singularity. A summary of local dexterity measures can be found in [5]. Since being introduced, local dexterity measures have been used effectively in mechanism design, path planning, and control applications [6], [7], [8], [9], [10]. Vahrenkamp et al. showed how a global manipulability distribution can be generated via random sampling to assess the robot's dexterity across the entire workspace [11]. Similar global dexterity indices have been introduced for kinematic [12] and dynamic [13] optimization of 3DOF parallel manipulators, showing how global dexterity measures hold great utility in optimal manipulator design and control. However, evaluating the Jacobian at every possible joint configuration presents a computational bottleneck that has prevented manipulability from being used more widely for global dexterity analysis.

As an alternative to using the manipulability measure for assessing dexterity, other work has generated capability maps via global workspace overview [14], [15]. The reachability set, or workspace, of a multibody kinematic chain, such as a robotic manipulator, is defined as the set of end-effector states that satisfy kinematic constraints, thus can be reached by the end-effector [2]. By hierarchically discretizing the task space, $SE(3)$, into voxels in Cartesian Space, $\mathbb{R}^3$, each with an associated rotational grid that discretizes $SO(3)$, dexterity can be evaluated by the percentage of bins that are reachable within a given voxel [16]. This approach effectively represents the workspace as a density function, establishing a link between redundancy, dexterity, and robustness to failure. Still, computing the reachability map requires an exhaustive forward kinematics assessment across the entire joint space. Alternatively, random joint sampling can be used to generate a wide range of end-effector samples in a short amount of time, but it does not guarantee that the entire workspace is covered in finite time.

One approach that has been taken to reduce the computational burden of random sampling is the Convolution Method [17], [18]. In this approach, the workspace is discretized into voxels. Then, each joint angle is randomly sampled and a

This material is based upon work supported by the National Science Foundation (NSF) Graduate Research Fellowship Program (Grant No. DGE1255832) and the NASA Space Technology Graduate Research Opportunities fellowship (Grant No. 80NSSC24K1371). Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF or NASA.

[1]Nate S. Osikowicz is a NASA Space Technology Graduate Research Fellow in the department of Aerospace Engineering at the Pennsylvania State University, University Park, Pennsylvania, USA nko5048@psu.edu

[2]John R. Cooper is with the Autonomous Integrated Systems Research Branch, NASA Langley Research Center, Hampton, Virginia, USA john.r.cooper@nasa.gov

[3]Puneet Singla is a Harry and Arlene Schell Professor of Engineering in the department of Aerospace Engineering at the Pennsylvania State University, University Park, Pennsylvania, USA psingla@psu.edu

histogram is generated for each link with bins representing the number of samples in each voxel. Next, the end-effector workspace is computed by recursively convolving the intermediate workspace densities of each link. Thus, for a manipulator with n joints, the end-effector workspace is computed by taking $n-1$ convolution integrals, greatly simplifying the computational burden of random sampling. In recent years, this approach has been further streamlined by taking the Fourier Transform of the workspace probability density function (PDF), which results in the workspace characteristic function. From the convolution theorem, n convolutions can be computed as the product of n characteristic functions. Aside from simplifying the convolution integral, the characteristic function representation of a PDF offers key insights into its underlying structure. Mainly, the Taylor Series Expansion of the characteristic function yields the Moment Generating Function (MGF) [19], which decomposes the PDF into its statistical moments. Thus, storing statistical moments of the PDF can be thought of as storing the spectral signature of a periodic function which, in general, is more efficient than retaining the entire function itself.

Guided by the Moment Generating Function, this paper investigates efficient methods for computing and propagating statistical moments to approximate the manipulator workspace density for path planning applications. One of the major technical hurdles lies in computing the workspace statistical moments in a computationally efficient manner. To overcome this hurdle, a quadrature scheme known as the Conjugate Unscented Transform (CUT) [20] is used to compute statistical moments with a minimum number of quadrature points. The main benefit of using CUT is that odd-order moment constraint equations (MCEs) are automatically satisfied due to symmetric structures. In addition to computing the workspace moments, another technical hurdle lies in the fact that approximating a PDF from its statistical moments is not a trivial task. One method computes mean and covariance in exponential coordinates [21] to save computational expense without sacrificing approximation accuracy. The method of computing and propagating moments has also been explored extensively in the polymers literature [22].

In this paper, an alternative method is presented that finds an optimal Gaussian Mixture Model (GMM) subject to the workspace moments. This builds upon previous work which used GMMs to approximate workspace densities [23]. However, the computational burden is further alleviated by posing the workspace approximation problem as a nonlinear optimization problem. This is accomplished by minimizing the two-norm error between the workspace moments and the GMM model moments, which are derived from the tensor-based framework introduced in [24]. One of the major benefits of using a mixture model to approximate the workspace is that mixture components can be added or subtracted from the model, allowing the user to trade approximation accuracy for computation time. This benefit is further enhanced by introducing an adaptive splitting algorithm that automatically satisfies the first two statistical moments in the mixture model.

Once the workspace density is generated, it can be utilized as an objective function in a sampling-based planner to maximize dexterity in manipulation tasks. Sample-based planning algorithms are a common approach to robot path planning. Popular methods include variations of the rapidly-exploring random trees (RRT) [25] and probabilistic roadmap (PRM) algorithms [26]. Optimal path planning methods, such as RRT* and PRM*, are used to generate paths that minimize an objective function [27].

The remainder of the paper is organized as follows: Section II defines the path planning problem in terms of the optimal dexterity cost function. Section III presents the solution methodology in two parts. First, the workspace density approximation is solved by finding the optimal Gaussian Mixture Model components subject to the workspace statistical moments. Then, a path planning algorithm is derived that maximizes the workspace density between the manipulator start and goal states. In Section IV, the algorithm is experimentally validated on the Precision Assembled Space Structure (PASS) subscale platform at NASA.

II. PROBLEM STATEMENT

Let $\vec{\theta}$ be a state in the joint space of a serial kinematic chain, and let $\vec{x}$ be a state in the corresponding workspace. $\vec{\theta}$ and $\vec{x}$ are related through the kinematics function $\vec{x} = f(\vec{\theta})$. A path through the joint space to a desired end-effector state, $\mathcal{P}$, is a sequence of n joint states starting at $\vec{\theta}_0$ and ending at a final state $\vec{\theta}_n$ which satisfies the goal criterion

$$||\vec{x}_g - f(\vec{\theta}_n)||_2 \leq \epsilon \tag{1}$$

where x_g is the goal state in the workspace and $\epsilon > 0$ is the goal tolerance. Traditionally, the path cost is defined by adding the Euclidean distance between sequential states,

$$J_1(\mathcal{P}) = \sum_{i=1}^{n} ||\vec{\theta}_i - \vec{\theta}_{i-1}||_2. \tag{2}$$

In this case, the problem is to find a path, $\mathcal{P}^* = \operatorname{argmin} J_1(\mathcal{P})$, such that the goal criterion and all kinematic constraints are satisfied. The optimal path, $\mathcal{P}^*$, which minimizes (2) is the shortest path to the goal in the joint angle space.

In some planning scenarios, it may be more desirable to maximize the probability of the path, rather than simply minimize the distance to the goal. Intuitively, characteristics of the system dynamics are revealed by the growth or depletion of trajectory concentrations in the workspace. Growth in concentration, or increased density, defines regions of high dexterity where the trajectories accumulate. On the other hand, depletion of concentration in the workspace implies instability or singularity. For example, if the start and goal states are near kinematic singularities, the path that minimizes (2) may lack the degrees of freedom that are needed to orient the end-effector in certain directions. One way to incorporate the likelihood of a path into the planning objective is with the workspace density function. Let $p(\vec{x}) = p(f(\vec{\theta}))$ be a model of the workspace density

function that is normalized such that $0 \leq p(\vec{x}) \leq 1$. Because multiple joint states can lead to the same end-effector position, the workspace density function is maximized in regions of high dexterity and minimized in regions of low dexterity. The total probability of path $\mathcal{P}$ is defined as $p(\mathcal{P}) = p(\vec{x}_1)p(\vec{x}_2)\dots p(\vec{x}_n)$. The logarithm can be used to convert multiplication into addition as $\log\big(p(\mathcal{P})\big) = \log\big(p(\vec{x}_1)\big) + \log\big(p(\vec{x}_2)\big) + \dots + \log\big(p(\vec{x}_n)\big)$. As such, a new cost function can be defined as

$$J_2(\mathcal{P}) = \sum_{i=1}^{n} \Big[\gamma||\vec{\theta}_i - \vec{\theta}_{i-1}||_2 - (1-\gamma)\log\big(p(\vec{x}_i)\big)\Big], \quad (3)$$

where $0 \leq \gamma \leq 1$ is a weight that can be used to balance path length with path dexterity or likelihood. By incorporating the normalized workspace density into (3), regions of high dexterity $\big(p(x) \simeq 1\big)$ result in a low J_2 cost while regions of low dexterity $\big(p(x) \simeq 0\big)$ result in a high J_2 cost. Then, the optimal dexterity path planning problem is to find a path, $\mathcal{P}^* = \operatorname{argmin} J_2(\mathcal{P})$ such that (1) and all kinematic constraints are satisfied. In this vein, Section III investigates how to compute an approximate model for the workspace density function with low computational expense. Then, it is shown how the workspace density approximation can be incorporated into a sampling based path planning algorithm.

III. METHODOLOGY

A. Workspace Density Approximation

To solve the problem in Section II, a suitable model for the workspace density function must be constructed. Given an n-link manipulator with link lengths $\vec{\ell}$ and joint angles $\vec{\theta}$ drawn from the uniform joint angle distribution $\mathcal{U}(\vec{\theta}_{min}, \vec{\theta}_{max})$, the reachability set is defined as

$$\mathcal{R}(\vec{\ell}, \mathcal{U}) = \vec{x}_j = \{f(\vec{\ell}, \vec{\theta}_j) \; \forall \; \vec{\theta}_j \in \mathcal{U}, j \in \mathbb{Z} \mid 0 \leq j \leq N\}$$

where N is the number of random samples and $\vec{x}_j = f(\vec{\ell}, \vec{\theta}_j)$ is the forward kinematics function that maps joint state $\vec{\theta}_j$ to end-effector state $\vec{x}_j$. That is, the reachability set is the forward mapping from all possible joint angles to all possible end-effector states. The workspace density problem is to compute the distribution of all possible end-effector states in the reachable set, $\mathcal{R}$, given knowledge of the link lengths and joint angle distribution, $\mathcal{U}(\vec{\theta}_{min}, \vec{\theta}_{max})$.

1) Forward Kinematics: The forward kinematics of a robotic manipulator can be derived in terms of any of the commonly used parameterizations (e.g DH parameters, URDF parameters, etc.). In this paper, Direction Cosine Matrices (DCMs) are used to express the end-effector position vector in terms of the link lengths and joint angles. To begin, the i^{th} link can be described by its length, ℓ_i, and angle, θ_i, with respect to the previous link. By attaching a reference frame to each of the links, the end-effector position can be computed through successive matrix multiplications. The vector along the i^{th} link is expressed in the i^{th} frame as $\vec{r}_i^{\,i} = [0, \ell_i]^T$. Similarly, the i^{th} link vector can be expressed in the global coordinate frame, $\mathbb{N}$, as $\vec{r}_i^{\,\mathbb{N}} = [x_i, y_i]^T$. Using this methodology, the i^{th} link can be converted from its local coordinate frame (frame i) to the parent frame (frame $i-1$) by multiplying by the DCM $C(\theta_i)$ as

$$\vec{r}_i^{\,i-1} = \underbrace{\begin{bmatrix}\cos(\theta_i) & -\sin(\theta_i)\\ \sin(\theta_i) & \cos(\theta_i)\end{bmatrix}}_{C(\theta_i)} \underbrace{\begin{bmatrix}0\\ \ell_i\end{bmatrix}}_{\vec{r}_i^{\,i}}. \quad (4)$$

Using the above methodology, the forward kinematics for an n-link serial manipulator are computed as

$$\vec{r}_n^{\,\mathbb{N}} = \sum_{i=1}^{n}\Big(\prod_{j=1}^{i} C(\theta_j)\Big)\vec{r}_i^{\,i}. \quad (5)$$

2) Workspace Statistical Moments: Equation (5) is a forward mapping from the joint angle space to the end-effector workspace. Therefore, the workspace density could be approximated by taking N random samples of the n-dimensional joint space and propagating them through (5) to approximate the density function with an appropriately binned histogram. However, this random sampling approach suffers from the curse of dimensionality due to the exponential growth in the number of end-effector samples, N^n. Therefore, random sampling is not a computationally feasible approach to workspace density approximation. As an alternative, the workspace density function can be approximated by its statistical moments as shown in [23]. It will now be shown how (5) can be used to derive a recursive formula for computing the d^{th}-order statistical moment of the end-effector. For clarity, only the first two moments, the mean and covariance, are shown here. For an n-link manipulator, this requires n steps. In the first step, the mean and covariance of the n^{th} (most distal) link are computed as follows. The mean of the n^{th} link is evaluated in its parent frame (frame $n-1$) as

$$\vec{\mu}_n = \mathbb{E}[\vec{r}_n^{\,n-1}] = \mathbb{E}[C(\theta_n)]\vec{r}_n^{\,n}. \quad (6)$$

Expanding (6) reveals how the expected values must be evaluated on the elements of $C(\theta_n)$:

$$\vec{\mu}_n = \mathbb{E}[\vec{r}_n^{\,n-1}] = \begin{bmatrix}\mathbb{E}[\cos(\theta_n)] & -\mathbb{E}[\sin(\theta_n)]\\ \mathbb{E}[\sin(\theta_n)] & \mathbb{E}[\cos(\theta_n)]\end{bmatrix}\begin{bmatrix}0\\ \ell_n\end{bmatrix}. \quad (7)$$

Because $\sin(\theta_n)$ and $\cos(\theta_n)$ are nonlinear in the joint variables, $\mathbb{E}[\sin(\theta_n)] \neq \sin(\mathbb{E}[\theta_n])$ and $\mathbb{E}[\cos(\theta_n)] \neq \cos(\mathbb{E}[\theta_n])$. Therefore, it can be concluded that a quadrature scheme must be employed to evaluate the element-wise expected values of a rotation matrix. Keeping this in mind, the covariance matrix of the n^{th} link is evaluated in its parent frame (frame $n-1$) similarly as

$$\Sigma_n = \mathbb{E}[C(\theta_n)]\vec{r}_n^{\,n}\vec{r}_n^{\,n^T}\mathbb{E}[C(\theta_n)^T] - \vec{\mu}_n\vec{\mu}_n^{\,T}. \quad (8)$$

In subsequent steps, the mean and covariance are updated by working distally from the end-effector to the base. The mean and 2nd non-central moment of the i^{th} link are evaluated in frame $i-1$ by using knowledge of the previously computed moments as

$$\begin{aligned}\vec{\mu}_i &= \mathbb{E}[C(\theta_i)](\vec{r}_i^{\,i} + \vec{\mu}_{i-1})\\ \Sigma_i &= \mathbb{E}[C(\theta_i)](\vec{r}_i^{\,i}\vec{r}_i^{\,i^T} + \vec{r}_i^{\,i}\vec{\mu}_{i-1}^{\,T}\\ &\quad + \vec{\mu}_{i-1}\vec{r}_i^{\,i^T} + \Sigma_{i-1})\mathbb{E}[C(\theta_i)^T].\end{aligned} \quad (9)$$

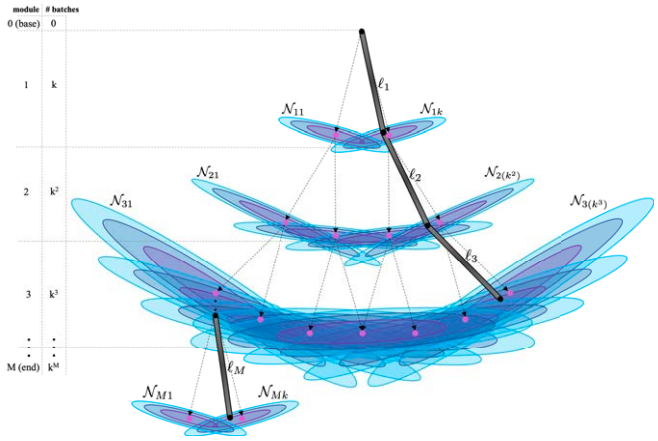


Fig. 1. Workspace approximation via Gaussian mixture model

Repeating this process for n links, a Gaussian approximation can be easily obtained for the end-effector distribution. The Gaussian workspace approximation is useful for rapid uncertainty analysis in a local domain, but it degrades as the joint angle bounds are increased. Therefore, higher-order statistical moments are required to more accurately capture the full nonlinear workspace distribution. To compute the higher-order tensor moments, an index notation must be introduced. Using index notation, (9) can be restated as

$$\begin{aligned} \mu_i^{\alpha} &= \mathbb{E}[C(\theta_i)^{\alpha a}]\big(\ell_i^a + \mu_{i-1}^a\big) \\ \Sigma_i^{\alpha\beta} &= \mathbb{E}[r_i^{\alpha} r_i^{\beta}] - \mu_i^{\alpha}\mu_i^{\beta}. \end{aligned} \tag{10}$$

where $\alpha = 1, 2, \cdots, r$, $a = 1, 2, \cdots, r$, and $\beta = 1, 2, \cdots, r$. The variable r corresponds to the dimension of the workspace (e.g. $r = 2$ for planar manipulators). Note that ℓ_i^a and μ_{i-1}^a represent the a^{th} component of the vectors $\vec{\ell}_i$ and $\vec{\mu}_{i-1}$, respectively. Similarly, $C(\theta_i)^{\alpha a}$ represents the αa component of the matrix $C(\theta_i)$. Notice that the index notation corresponds to summation and can be expanded as

$$\mu_i^{\alpha} = \mathbb{E}[C(\theta_i)^{\alpha a}]\big(\ell_i^a + \mu_{i-1}^a\big) = \sum_{a=1}^{r} \mathbb{E}[C(\theta_i)^{\alpha a}]\big(\ell_i^a + \mu_{i-1}^a\big)$$

Using a similar process, the third-order moment tensor is computed as

$$\begin{aligned} m_{3i}^{\alpha\beta\gamma} = \mathbb{E}[r_i^{\alpha} r_i^{\beta} r_i^{\gamma}] - \Big(&\mu_i^{\alpha}\mathbb{E}[r_i^{\beta} r_i^{\gamma}] + \mu_i^{\beta}\mathbb{E}[r_i^{\alpha} r_i^{\gamma}] \\ &+ \mu_i^{\gamma}\mathbb{E}[r_i^{\alpha} r_i^{\beta}]\Big) + 2\mu_i^{\alpha}\mu_i^{\beta}\mu_i^{\gamma}. \end{aligned} \tag{11}$$

The fourth-order and higher order moments are computed in a similar fashion. It should be noted that the expected values in (10)-(11) can be computed with any desired quadrature scheme. In this paper, the Conjugate Unscented Transform (CUT) is used for computing the expected values with a minimum number of quadrature points. More detail on the benefits of using CUT for workspace moment approximation can be found in [23].

3) Adaptive Gaussian Mixture Model from Higher Order Moments (AGM-HOM): In this section, the results from Section III-A.2 are used to approximate the workspace density function as a Gaussian Mixture Model,

$$p(\vec{x}) = \sum_{i=1}^{k} w_i \, \mathcal{N}_i\left(\vec{x} \middle| \mu_i, \Sigma_i\right), \tag{12}$$

where $\mathcal{N}_i$ represents the ith Gaussian component, w_i is its weight, and k is the total number of components in the mixture model. Previous work [23] generates a workspace mixture model by discretizing the joint angle space into k batches per joint and then applies (9) to each batch to compute the workspace mixture components. This yields a mixture model that scales exponentially in the number of components as the number of batches per joint increases. This exponential growth is illustrated in Fig. 1. To circumvent this curse of dimensionality, the higher-order moments can be leveraged to formulate an optimal mixture model. This is done by posing the mixture model as a nonlinear optimization problem:

$$\begin{aligned} &\text{minimize:} \quad ||\vec{m}(\vec{x}) - \vec{\mathcal{M}}(\vec{p})||_2 \\ &\text{subject to:} \quad 0 \le w_i \le 1, \ \sum_{i=1}^{k} w_i = 1, \ \Sigma_i \ge 0 \quad \forall\, i \in [1, k] \end{aligned} \tag{13}$$

where $\vec{m}(\vec{x})$ is the vector of workspace moments computed in (10)-(11) and $\vec{\mathcal{M}}(\vec{p})$ is the vector of Gaussian Mixture Model moments, which are a function of the mixture parameters contained in the vector $\vec{p} = [\mu_1, \mu_2, \cdots, \mu_k, \Sigma_1, \Sigma_2, \cdots, \Sigma_k, w_1, w_2, \cdots, w_k]$. It is noted that the d^{th} order moment of a Gaussian element is computed as

$$\mathcal{M}(\vec{p})_d = \sum_{i=0}^{d/2} \binom{d}{2i} \frac{2i!}{i!2^i} \text{sym}\big(\vec{\mu}^{(\otimes d-2i)} \otimes \Sigma^{\otimes i}\big) \tag{14}$$

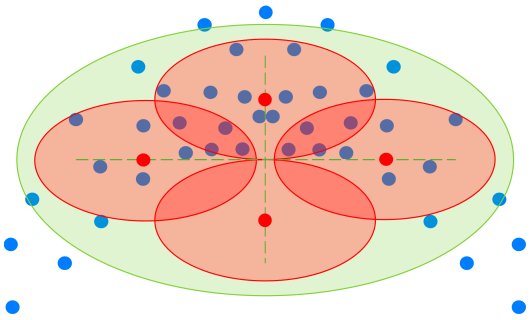
where sym() is the tensor symmetrization [24].

4) Adaptive Splitting Algorithm: One of the challenges of solving the optimization problem in (13) is choosing the initial condition for the mixture parameters. Another challenge is deciding how many mixture components to include in the model. To alleviate both of these challenges, a splitting algorithm that automatically satisfies the first and second order moments is adapted from [28]. In this approach, illustrated in Fig. 2, $2r$ mixture components are initially chosen to approximate the r-dimensional workspace as

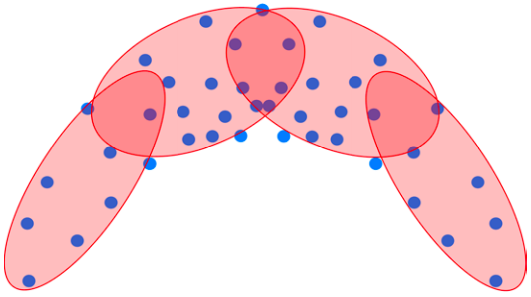
$$\vec{\mu} = \sum_{i=1}^{2r} w_i \vec{\mu}_i, \tag{15}$$

$$\Sigma = \sum_{i=1}^{2r} w_i [\Sigma_i + \vec{\mu}_i \vec{\mu}_i^T], \tag{16}$$

$$w_i = \frac{1}{2r} \ \forall i \in [1 : 2r]. \tag{17}$$



(a) Approximation of the original component with $2r$ components



(b) Propagation of $2r$ components via AGM-HOM

Fig. 2. Illustration of the AGM-HOM splitting and propagation steps

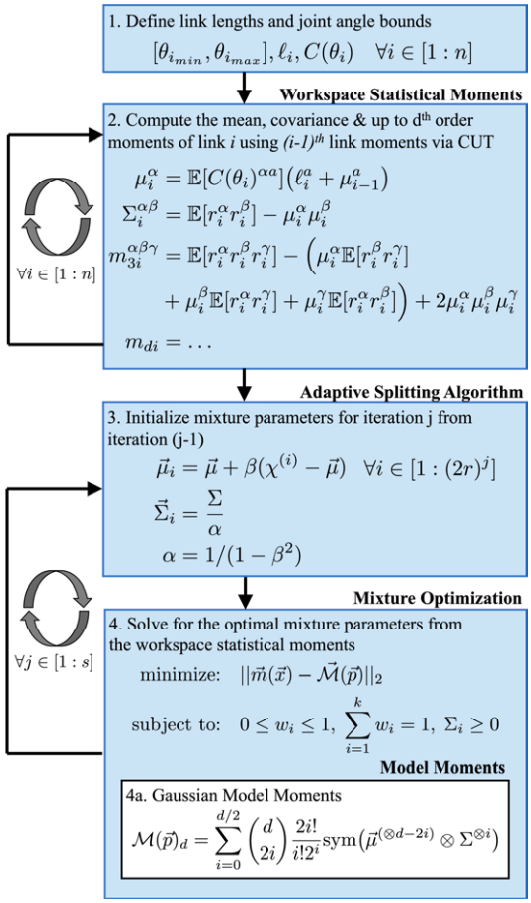


Fig. 3. Flow Chart of the AGM-HOM algorithm

. The splitting components are initialized by taking the Unscented Transform (UT) [20] of the covariance ellipsoid that is defined by the 2nd workspace moment, Σ, and centered at the 1st workspace moment (the mean), $\vec{\mu}$. Let $\chi^{(i)}$ be the i^{th} sigma point defined in [20]. Then, the model parameters are initialized as

$$\vec{\mu}_i = \vec{\mu} + \beta(\chi^{(i)} - \vec{\mu}) \; \forall i \in [1:2r], \tag{18}$$

$$\vec{\Sigma}_i = \frac{\Sigma}{\alpha} \; \forall i \in [1:2r], \tag{19}$$

$$\alpha = \frac{1}{1-\beta^2}, \tag{20}$$

where $\beta < 1$ is a free parameter which determines the spread of the new components around the original mean and α controls the shrinkage of the new components with respect to the original component. By substituting (18)-(20) into (15)-(17), it can be seen how the adaptive splitting algorithm automatically satisfies the first two moment constraints. Once the initial $2r$ components are placed, the optimization algorithm in (13) is solved for the optimal mixture components. If further approximation accuracy is desired, the splitting algorithm can be re-iterated on the optimized Gaussian components to yield more components.

5) Algorithm Overview: Fig. 3 summarizes how the contributions from Sections III-A.2-III-A.4 can be combined to yield a powerful framework for workspace density approximation known as the Adaptive Gaussian Mixture Model from Higher-Order Moments (AGM-HOM). In Step 1, the manipulator parameters (link lengths and joint angle bounds) are specified and then used in Step 2 to compute the workspace statistical moments via the recursive approach presented in (10)-(11). Once the workspace moments are computed, the workspace mean and covariance can be used to initialize the Adaptive Splitting Algorithm in Step 3. Fig. 2(a) illustrates Step 3 with initial mixture components shown in red. Representative Monte Carlo samples are shown in blue. The initial components are placed along the major and minor axes of the workspace covariance ellipsoid shown in green. Then, the optimal Gaussian Mixture Model components are solved in Step 4 to yield the new workspace density approximation shown in Fig. 2(b). The algorithm is reiterated for s iterations of adaptive splitting to yield a final mixture model with $k = (2r)^s$ mixture components.

B. Motion Planning

The RRT* algorithm is implemented to solve the path optimization problem defined in Section II. A search tree, T, is constructed with a root node of $\vec{\theta}_0$. The edge cost from a node at $\vec{\theta}_i$ to a node at $\vec{\theta}_j$ is

$$c(\vec{\theta}_i, \vec{\theta}_j) = \gamma ||\vec{\theta}_j - \vec{\theta}_i||_2 - (1-\gamma)\log\big(p(\vec{x}_j)\big) \tag{21}$$

where $0 \le \gamma \le 1$ and $p(\vec{x}_j)$ is the workspace density function evaluated at $\vec{x}_j = f(\vec{\theta}_j)$. To add nodes to the tree, joint states are sampled uniformly. The parent node is chosen

to be the node in the tree with minimal edge cost,

$$\vec{\theta}_p = \underset{\vec{\theta}_t \in T}{\operatorname{argmin}}\, c(\vec{\theta}_t, \vec{\theta}_s), \tag{22}$$

where $\vec{\theta}_p$ is the state of the selected parent node, $\vec{\theta}_s$ is the randomly sampled state and $\vec{\theta}_t$ is the set of states already in T. A new node is then generated by taking a step from $\vec{\theta}_p$ towards $\vec{\theta}_s$ with step size, δ,

$$\vec{\theta}_{new} = \vec{\theta}_p + \frac{\vec{\theta}_s}{||\vec{\theta}_s||}\delta. \tag{23}$$

If the motion from $\vec{\theta}_p$ to $\vec{\theta}_{new}$ satisfies all kinematic constraints, $\vec{\theta}_{new}$ is added to T with an edge from $\vec{\theta}_p$. Otherwise, the process is repeated with a new randomly sampled state.

Each time a new node is added to T, the tree is rewired to minimize the total path cost. For each node in a neighborhood around $\vec{\theta}_{new}$, the edge cost from that node's current parent is compared to the edge cost from $\vec{\theta}_{new}$. If traveling from $\vec{\theta}_{new}$ results in a lower edge cost, the parent of the node is changed to be $\vec{\theta}_{new}$.

As the number of iterations approaches infinity, the sum of edge costs to travel from the root node to any other node converges to the true optimum. In practice, the algorithm is run until a maximum tree size, maximum number of iterations, or maximum run time occurs. For any node satisfying the goal criterion in (1), a path is found by traversing backwards through T to the root node.

Once an optimal path is found, a C^3 continuous polynomial spline is fit to pass through the resulting waypoints from the path planning algorithm with the first three derivatives constrained to 0 at the start and end points. The spline is used to generate joint velocity commands for the robot via a virtual leader approach based on that from [29]. Let the spline be represented by $\vec{\theta}_l(s)$ where $s \in [0, n]$ for n spline segments. $\vec{\theta}_l(0) = \vec{\theta}_0$ and $\vec{\theta}_l(n) = \vec{\theta}_g$. Integer values for s represent the points of transition between spline segments. s evolves according to

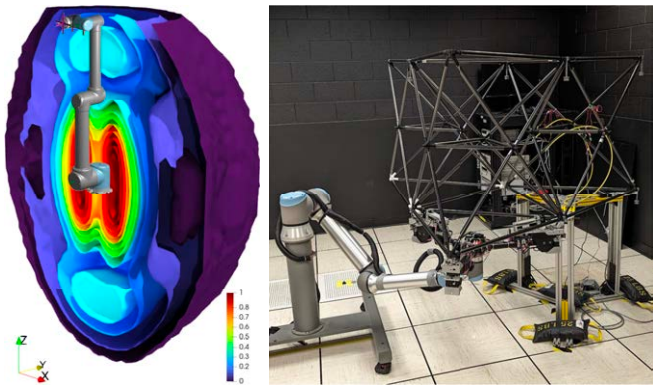
$$\ddot{\hat{s}} = \delta(n - s) + \epsilon||\vec{\theta}_l(s) - \vec{\theta}||, \tag{24}$$

$$\dot{\hat{s}} = \begin{cases} s_{d_{min}}, & \text{if } \dot{\hat{s}} < s_{d_{min}} \\ s_{d_{max}}, & \text{if } \dot{\hat{s}} > s_{d_{max}} \\ \dot{\hat{s}} & \text{otherwise} \end{cases}, \tag{25}$$

$$s = \begin{cases} 0, & \text{if } \bar{s} < 0 \\ n, & \text{if } \bar{s} > n \\ \bar{s} & \text{otherwise} \end{cases}, \tag{26}$$

where δ is a gain driving the virtual leader forward along the spline, ϵ is a gain that slows the virtual leader down as its distance from the real robot increases, $\vec{\theta}$ is the current robot joint state, and $s_{d_{min}}$ and $s_{d_{max}}$ are the minimum and maximum values for the rate of change of s respectively. Joint velocity commands, $\vec{v}$, are generated according to

$$\vec{v} = \dot{\vec{\theta}}_l + K(\vec{\theta}_l - \vec{\theta}), \tag{27}$$



(a) UR10e workspace density (b) Truss assembly goal state

Fig. 4. Precision Assembled Space Structure (PASS) Subscale platform

where $\dot{\vec{\theta}} = \frac{d\vec{\theta}}{ds}\dot{s}$ is a feedforward term to match the virtual leader's velocity, and K is a proportional feedback gain to account for errors in tracking the virtual leader.

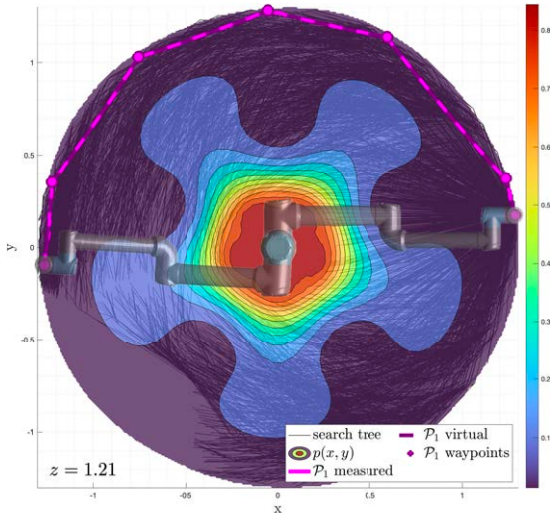
IV. EXPERIMENTAL VALIDATION

To validate the algorithm, the workspace density was utilized in a path planning scenario on NASA's Precision Assembled Space Structure (PASS) Subscale Platform [30]. Figure 4(b) shows the experimental platform comprising a UR10e manipulator and three tritruss structures. The objective is to find a collision-free joint sequence to move a tritruss structure from the starting configuration at $\vec{\theta}_0 = [0\ \ 0\ \ -15\ \ -75\ \ \ 90\ \ \ 225]^\circ$ to the truss placement region defined by $||\vec{x}_g - \vec{x}_n||_2 \leq 0.1$ where $\vec{x}_g = [x_g\ \ \ y_g] = [-1.28\ \ \ -0.17]$ meters. Additionally, the end-effector is desired to remain upright (orientation fixed) and stay in the motion plane defined by the truss stand height at $z = 1.21$ meters. To account for this constraint, the RRT* search tree is constructed in the end-effector xy-plane with collision-checking between nodes done via inverse kinematics,

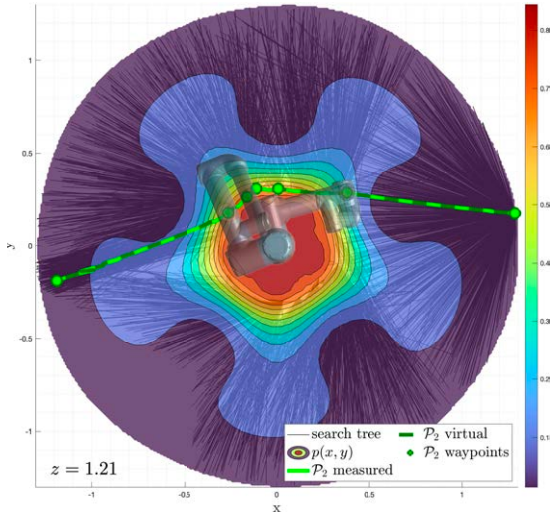
$$\vec{\theta}_{i+1} = \vec{\theta}_i + \left(J^T(\theta_i)J(\theta_i) + \lambda\right)^{-1} J^T(\theta_i)\delta\vec{x}_i, \tag{28}$$

where $\lambda = 1E-3$, $J(\theta)$ is the manipulator Jacobian, and $\delta\vec{x}_i$ is a vector pointing from $\vec{x}_i$ to $\vec{x}_{i+1}$ with length $\ell_\delta = 0.01$ specifying the collision checking resolution. Because collisions are accounted for directly in the path planning algorithm, the workspace density can be generated across the entire range of motion and then evaluated only at collision-free states to compute the cost in (21). As such, the UR10e workspace density was approximated from DH parameters provided by the manufacturer and uniformly distributed joint angles $\vec{\theta} \sim \mathcal{U}(-180^\circ, 180^\circ)$, as illustrated in Figure 4(a).

Next, two different scenarios were implemented to show the utility of the modified cost function. In the first scenario, the J_1 cost function (2) was implemented to yield the magenta path shown in Fig. 5(a). Next, the J_2 cost function (3) was implemented with $\gamma = 0.3$ to yield the green path shown in Fig. 5(b). It is illustrated how the modified cost function pulls the end-effector into the higher density region as (3) is minimized. This behavior is due to the second term



(a) Search tree and optimal path from J_1 cost minimization



(b) Search tree and optimal path from J_2 cost minimization

Fig. 5. Comparison of two different path planning scenarios implemented on the PASS Subscale Platform (overhead view)

in (3), which has the effect of decreasing the cost of end-effector states that lie in high workspace density regions. In both cases, parameters for the virtual leader controller are as follows: K is a 6×6 diagonal matrix with all diagonal elements 0.25 except $K_{6,6} = 0$, $\delta = 0.1$, $\epsilon = 1$, $s_{d_{min}} = -1$, $s_{d_{max}} = 1$. Figures 6(a)-6(d), which compare the virtual leader and recorded joint data, confirm that the manipulator follows the waypoints as expected. Fig. 6(e) illustrates that $J_1(\mathcal{P}_1)$ and $J_2(\mathcal{P}_2)$ decrease monotonically as the search tree is expanded for 5000 iterations, showing the asymptotic optimality of the path planning algorithm. Moreover, it is shown that $J_2(\mathcal{P}_1)$ is roughly 2.5x larger than $J_2(\mathcal{P}_2)$ after 5000 iterations, confirming that $\mathcal{P}_2$ is indeed the higher dexterity path.

V. CONCLUSION

This paper has demonstrated how the workspace density function can be approximated with low computational cost and then used to plan maximum dexterity paths for robotic manipulators. Sections II-III introduced the AGM-HOM algorithm for approximating the manipulator workspace density from statistical moments. In this vein, the Conjugate Unscented Transform (CUT) was combined with an adaptive splitting algorithm to approximate the workspace density with a minimal number of quadrature points. Then, a new cost function was developed that uses the workspace density to maximize manipulator dexterity in path planning applications. In Section IV, the algorithm was implemented on the Precision Assembled Space Structure (PASS) subscale platform at NASA. First, the AGM-HOM algorithm was used to approximate the workspace density of the UR10e manipulator. Then, it was demonstrated how the newly developed cost function effectively increased the manipulator dexterity when compared to the traditional RRT* cost function. Given this success and the generality of the underlying methods, the authors believe that the resulting algorithm could be utilized in a wide array of dexterous manipulation tasks.

REFERENCES

[1] F. Park and R. Brockett, "Kinematic dexterity of robotic mechanisms." International Journal of Robotics Research, 1993, pp. 1–15.

[2] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge University Press, 2017.

[3] T. Yoshikawa, "Manipulability of robotic mechanisms." International Journal of Robotics Research, 1985.

[4] J. Salisbury and J. Craig, "Articulated hands: Force control and kinematic issues." International Journal of Robotics Research, 1982.

[5] J.-O. Kim and P. Khosla, "Dexterity measures for design and control of manipulators." IEEE International Workshop on Intelligent Robots and Systems (IROS), 1991.

[6] S. Tadokoro, I. Kimura, and T. Takamori, "A dexterity measure for trajectory planning and kinematic design of redundant manipulators." 15th Annual Conference of IEEE Industrial Electronics Society, Philadelphia, PA, USA, 1989.

[7] M. Azad, J. Babi, and M. Mistry, "Effects of the weighting matrix on dynamic manipulability of robots." Autonomous Robots, 2019.

[8] L. Petrović, F. Marić, I. Marković, J. Kelly, and I. Petrović, "Trajectory optimization with geometry-aware singularity avoidance for robot motion planning." The 21st International Conference on Control, Automation and Systems (ICCAS 2021), 2021.

[9] J. Haviland and P. Corke, "A purely-reactive manipulability-maximising motion controller." arXiv preprint arXiv:2002.11901, 2020.

[10] Y. Shen, Y. Hong, W. Zhou, R. Tai, Y. Yuan, and H. Ding, "Manipulability and robustness optimization of the cable-driven redundant soft manipulator." Proceedings of the 2021 IEEE International Conference on Robotics and Biomimetics, 2021.

[11] N. Vahrenkamp, T. Asfour, G. Metta, G. Sandini, and R. Dillmann, "Manipulability analysis." IEEE-RAS International Conference on Humanoid Robots, 2012.

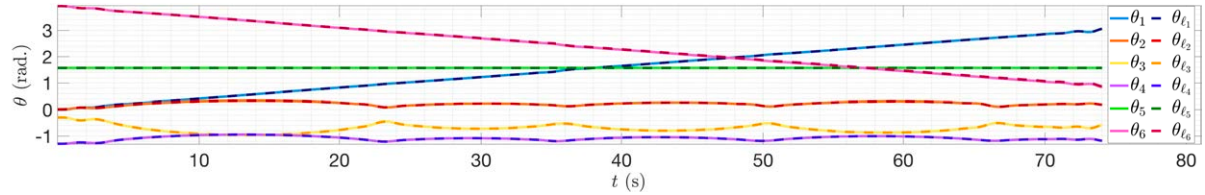
[12] C. Gosselin and J. Angeles, "A global performance index for the kinematic optimization of robotic manipulators." ASME Journal of Mechanical Design, Volume 113, Issue 3, 1991.

[13] L. G. H. Shao, L. Wang and J. Wu, "Dynamic manipulability and optimization of a redundant three dof planar parallel manipulator." 2009 ASME/IFToMM International Conference on Reconfigurable Mechanisms and Robots, 2009.

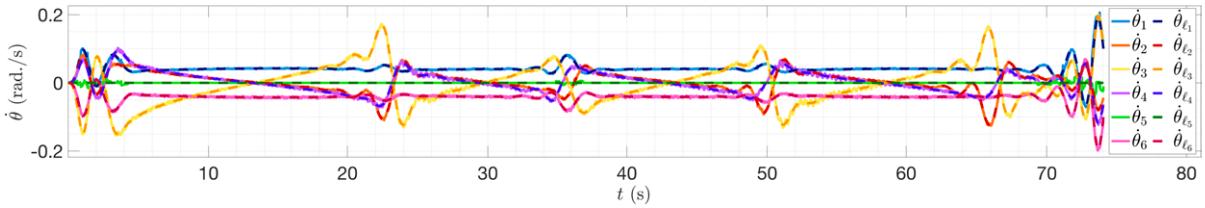
[14] O. Porges, D. Leidner, and M. Roa, "Planning fail-safe trajectories for space robotic arms." Frontiers in Robotics and AI, November 2021, pp. Volume 8, Article 710 021.

[15] O. Porges, R. Lapariello, J. Artigas, A. Wedler, C. Borst, and M. Roa, "Reachability and dexterity: Analysis and applications for space robotics," in *German Aerospace Center (DLR), 82234 Wessling, Germany*.

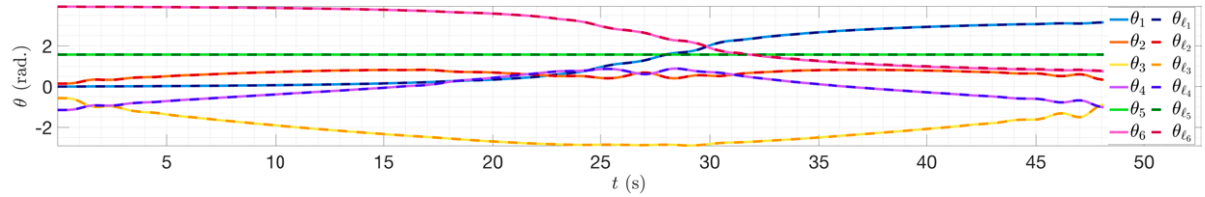
[16] F. Zacharias, "Knowledge representations for planning manipulation tasks." Vol. 16. Berlin, Germany: Springer, 2012.



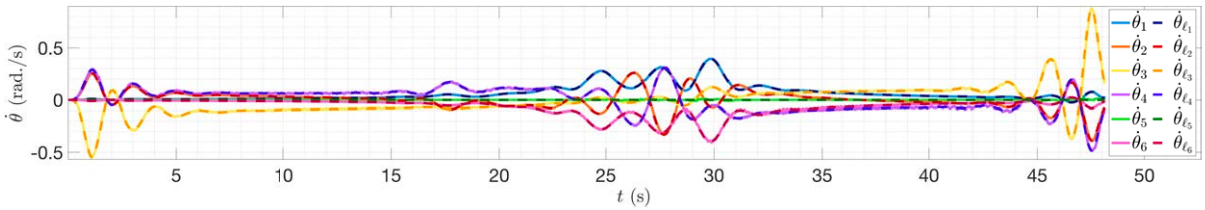
(a) Joint angles for J1 trajectory



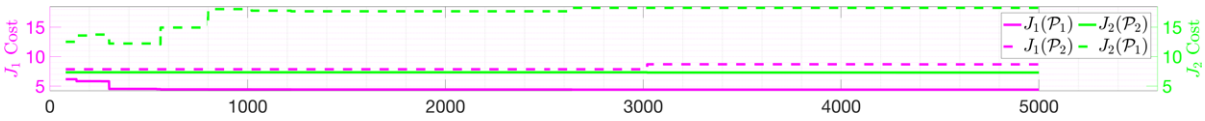
(b) Joint rates for J1 trajectory



(c) Joint angles for J2 trajectory



(d) Joint rates for J2 trajectory



(e) Path cost vs. iteration number for J1 and J2 search trees

Fig. 6. Comparison of virtual leader, recorded joint data, and path costs for J1 and J2 trajectories

[17] I. Ebert-Uphoff and G. S. Chirikjian, "Numerical convolution on the euclidean group with applications to workspace generation," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 1, 1998.

[18] J. Suthakorn and G. Chirikjian, "A new inverse kinematics algorithm for binary manipulators with many actuators," *Advanced Robotics*, vol. 15, no. 2, pp. 225–244, 2001.

[19] A. Jain, D. Gueho, and P. Singla, "A computationally efficient approach for stochastic reachability set analysis," in *AIAA Scitech 2020 Forum*, 2020.

[20] N. Adurthi, P. Singla, and T. Singh, "The Conjugate Unscented Transform- An approach to evaluate multi-dimensional expectation integrals," in *Proceedings of the American Control Conference*, 2012.

[21] A. Long, K. Wolfe, M. Mashner, and G. Chirikjian, "The banana distribution is gaussian: A localization study with exponential coordinates," *Robotics: Science and Systems VIII*, vol. 265, no. 1, 2013.

[22] M. Rehahn, W. Mattice, and U. Suter, *Rotational isomeric state models in macromolecular systems*. Springer Advances in Polymer Science, 2006.

[23] N. Osikowicz and P. Singla, "A gaussian mixture model for probabilistic workspace generation of multibody systems." IEEE Aerospace Conference, Big Sky, MT, 2025.

[24] J. K. J. Pereira and T. Kolda, "Tensor Moments of Gaussian Mixture Models: Theory and applications," in *arXiv:2202.06930v2 [stat.ML]*, 21 Mar 2022.

[25] S. M. LaValle and J. J. Kuffner Jr, "Randomized kinodynamic planning," *The international journal of robotics research*, vol. 20, no. 5, pp. 378–400, 2001.

[26] L. E. Kavraki, P. Svestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[27] S. Karaman, M. R. Walter, A. Perez, E. Frazzoli, and S. Teller, "Anytime motion planning using the RRT," in *2011 IEEE International Conference on Robotics and Automation*. IEEE, 2011, pp. 1478–1483.

[28] K. Vishwajeet and P. Singla, "Adaptive splitting technique for gaussian mixture models to solve Kolmogorov equation," in *American Control Conference*, 2014.

[29] D. L. Bacher, J. R. Cooper, and J. P. Navarro, "Trajectory generation with load constraints for robotic manipulators," in *ASCEND*. AIAA, 2023.

[30] J. R. Cooper, J. R. Martin, O. R. Stohlman, C. J. Cresta, T. V. Avila, R. Rajaram, and A. K. McQuarry, "Test results for autonomous assembly of modular space structures," in *ASCEND*. AIAA, 2023.