IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.xxxx.DOI

Heterogeneous Dimensionality Reduction for Efficient Motion Planning in High-dimensional Spaces

HUAN YU[1],WENJIE LU[2],YONGQIANG HAN[1],DIKAI LIU[2], MIAO ZHANG[1]
[1]School of Automation, Beijing Institute of Technology, Beijing, 100081 China
[2]Centre for Autonomous Systems, University of Technology Sydney, NSW, 2007, Australia

Corresponding author: Wenjie Lu (wenjie.lu@outlook.com).

This work was supported in part by the Australian Research Council (ARC) Linkage Project (LP150100935), the Roads and Maritime Services of NSW, the Centre for Autonomous Systems (CAS) at the University of Technology Sydney, and China Scholarship Council (CSC).

ABSTRACT Increasing the dimensionality of the configuration space quickly makes trajectory planning computationally intractable. This paper presents an efficient motion planning approach that exploits the heterogeneous low-dimensional structures of a given planning problem. These heterogeneous structures are obtained via a Dirichlet process (DP) mixture model and together cover the entire configuration space, resulting in more dimensionality reduction than single-structure approaches from the existing literature. Then, a unified low-dimensional trajectory optimization problem is formulated based on the obtained heterogeneous structures and a proposed transversality condition which is further solved via SQP in our implementation. The positive results demonstrate the feasibility and efficiency of our trajectory planning approach on an autonomous underwater vehicle (AUV) and a high-dimensional intervention autonomous underwater vehicle (I-AUV) in cluttered 3D environments.

INDEX TERMS Motion planning, Underwater vehicle, Trajectory optimization, Dimensionality reduction

I. INTRODUCTION

Robots designed for performing complex tasks, such as (mobile) manipulators and humanoid robots, often have many degrees of freedom (DOFs). Trajectory planning is an essential part of task execution. Compared with low-DOF robots, planning for high-DOF robots is challenging due to the high dimensionality of the configuration space. In this work, we use the trajectory planning problem of a high-DOF I-AUV as an example to introduce a new kinematic motion planning approach. The I-AUV consists of two parts, a 6-DOF base and a n-DOF manipulator that can be applied to conduct underwater infrastructure cleaning, inspection or data collection [1]. Our method can also be applied to other high-dimensional planning problems [2], [3].

In high-DOF robot planning, considering the computational time and memory required to solve an optimal trajectory, planning in high dimensional space is generally a computationally hard problem [3]. Although it is necessary to plan every DOF of the robot for the controller, it may not be necessary to search the trajectory throughout the entire configuration space because in some cases, the decision space may be small, and the planned trajectories retain low-dimensional structures [3]. For example, when planning a trajectory for an I-AUV in an environment with few obstacles, we need only to plan the base of the I-AUV; the manipulator trajectory could be easily decided by control methods [1]. Therefore, the high-dimensional decision space of the planning problem is reduced to a lower dimensional subspace. In other cases, only a few important DOFs may be needed to complete the task. The trajectory planned in lower dimensional subspace can easily be transformed into the original high-dimensional space. This process is called *planning with dimensionality reduction or low-dimensional structure* [3]. The low-dimensional structure is the basis of the low-dimensional subspace.

The lower dimensional subspace, such as the configuration space of the base of an I-AUV, can be designed manually based on prior knowledge. However, when given specific planning problem data, such as the cost function, obstacle population and the kinematic model of the robot, an ideal dimensionality reduction method is expected to automatically exploit the lower dimensionality space.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

In general cases, the lower dimensional subspace of the planning problem not obvious, requiring an unknown mapping from the original configuration space. For a given planning problem, the dimensionality reduction or low-dimensional structure learning process as shown in Section IV-A is to learn a low-dimensional structure that captures the cost function of the planning problem [4]. The high-dimensional cost function can be approximated by a low-dimensional cost function defined in the learned low-dimensional subspace.

The low-dimensional structure generally is learned from the cost structure or from experts' demonstration data based on various learning- or modeling-based methods [4], [5]. In those prior works, the dimesionality reduction process learns only a single low-dimensional structure. However, given a planning problem, the structure generally is heterogeneous. The low-dimensional structure is highly correlated to the environment information. In heterogeneous environments, considering different obstacle populations, obstacle-free regions, cluttered areas, narrow passages, etc. may all exist. Thus, in different regions, the dimensions and structures of the low-dimensional subspace to be considered to capture the original planning problem are obviously different. This fact is captured in the proposed heterogeneous dimensionality reduction in Section IV. In Section V-A, we demonstrate that heterogeneous dimensionality reduction is more efficient and effective.

Contributions: In this paper, rather than utilizing a *single-structure dimensionality reduction* method - planning in one low-dimensional subspace that is exploited by uniform dimensionality reduction [4], we propose a planning approach based on a heterogeneous dimensionality reduction approach. In this approach, the entire configuration space is decomposed into multiple sub-configuration spaces; then the planning problem has different low-dimensional structures in the different sub-configuration spaces. The trajectory is optimized in multiple subspaces based on the defined objective function and the learned low-dimensional structures. The main contributions of this work are as follows:

(i) *Identifiable low-dimensional subspace learning.* In our approach, trajectory optimization is performed in multiple low-dimensional subspaces. To identify the label of subspace that a low-dimensional point belongs, we propose a low-dimensional structure learning method as shown in Section IV-A1 that retains some dimensions of the original configuration space. These retained dimensions are often essential in distinguishing environment conditions, such as positional dimensions to distinguish obstacle influcence. By doing so, every point on the trajectory in the low dimensional spaces and every low dimensional structure retains some common dimensions. These retained dimensions identify an appropriate low dimensional structure for each point on the trajectory during planning. In this approach, low-dimensional structure is learned from the gradients of the cost function at sampled points via a spectral optimization method similar to principal component analysis (PCA).

(ii) *DP-based subspaces classification.* Because the distinct number of low-dimensional structures is unknown beforehand for a given map, a nonparametric clustering method based on the Dirichlet process is proposed to find all clusters and assign cluster labels to each sampled point. The DP is a stochastic process widely applied in Bayesian nonparametrics to cluster unclassified data. It is also widely used in motion pattern modeling [6]. In this work, we utilize it for planning spaces classification. Finally, an SVM-based classifier is trained based on those labeled points. When performing trajectory optimization, the subspace label of each every newly sampled point can be determined by the SVM classifier.

(iii) *Unified trajectory optimization in multiple subspaces.* Based on the objective function we defined, a unified gradient-based trajectory optimization is proposed. The trajectory planning process is divided into two steps. In the first step, optimization is processed in multiple low-dimensional subspace with a gradient-based optimization approach. The low-dimensional subspaces trajectory optimization method does not split the trajectory into multiple segments and optimize it in each subspace, but rather unifies these segments of heterogeneous dimensions through a proposed transversality condition. The second step is to map the optimized low-dimensional trajectory to full-dimensional trajectory with a sequential quadratic programming (SQP) solver. Our unified trajectory optimization problem analogous to the discrete switched systems control problem that is how to coordinate various subsystems [7]. Based on the defined optimization problem, the transversality condition and the switching equation in both steps are detailed in Section IV-B.

(iv) *Numerical experiments.* To verify the efficiency of the proposed approach, we first give the computational complexity analysis of our approach, and then apply our algorithm to a 3D AUV planning problem and a 10-DOF I-AUV planning problem. Numerical simulation results demonstrate the feasibility and efficiency of the proposed trajectory planning approach.

In the remainder of this paper, Section II introduces prior works regarding high-dimensional trajectory planning and planning methods with low-dimensional structures, and Section III defines the problem addressed in this work. Section IV details each module of our approach using 3D space motion planning task as an example. Section V reports the algorithm simulation results and its comparisons with other methods. This work is concluded in Section VI.

II. RELATED WORK

High-dimensional motion planning is a challenging topic. Next, we discuss the latest motion planning approaches applied to high-dimensional space planning problems and their relationships with our approach. We first review the latest generic planners, including search- or sampling-based methods, and optimization-based planners. Then, we focus on planning methods that utilize the low-dimensional structure of the planning problem.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

A. SEARCH- AND SAMPLING-BASED PLANNING

Search-based planners are the most widely applied methods for low-dimensional systems that guarantee optimality and completeness. However, it is extremely inefficient to discretize a high-dimensional configuration space, build a graph and then search that graph. Until recently, the typical approach for applying search-based approaches to high-dimensional systems was to use a lattice graph-based method [8], [9]. This approach means that the planner considers only limited possible robot actions and obtains a path with bounded suboptimality.

Sampling-based motion planning is the most popular approach for robot planning in high-dimensional configuration spaces. Sampling-based methods such as the probabilistic roadmap algorithm (PRM*), rapidly-exploring random trees (RRT*), and fast marching tree (FMT*) method guarantee probabilistic completeness and asymptotic optimality. In [10], FMT* was shown to have a higher convergence rate than PRM* and RRT*, which utilize a 'lazy' dynamic programming recursion method to generate a tree.

Although sampling-based methods are more efficient than search-based methods, the number of samples increases exponentially when the dimensionality increases and the sample density remains the same; therefore, the complexity increases at least exponentially. To improve the speed of sampling-based planners, as apposed to sampling the state space uniformly, one intuitive method is to perform sampling using heuristics. The batch informed trees (BIT*) method proposed in [11] utilize a heuristic strategy to adaptively change the area where dense sampling should be performed. The results in [11] demonstrate that BIT* is faster than RRT* and FMT* in some cases. Rather than using manually designed heuristics, recent work by Brian et al. [12] presented a more general non-uniform sampling strategy by using a supervised learning method.

The main idea of heuristic sampling is to reduce the search space or bias search priority. The method proposed in this paper also attempts to reduce the decision space by learning from problem data automatically.

B. OPTIMIZATION-BASED TRAJECTORY PLANNING

Optimization-based approaches formulate the planning problem as an optimization problem. They use an objective function to describe planning tasks, such as the shortest path or minimum energy use, and include many practical constraints. Then, the problem can be solved by an appropriate solver. The main advantage of these methods is that they utilize the derivative information, which can improve an initial trajectory to an optimal trajectory iteratively. Therefore, if the gradient can be solved accurately and efficiently, such methods may be more appropriate for application to high-dimensional planning problems. To enable online high-DOF robot planning, CHOMP proposed in [13] introduced an inexpensive method of calculating functional gradients by using a covariant update rule. When the planning task considers many general constraints, gradient information may not be available. Kalakrishnan et al. [14] applied a trajectory sampling strategy to solve the gradient approximately.

Differential dynamic programming, such as iLQG [15], forms another class of optimization-based methods that optimize the trajectory iteratively by solving a local quadratic programming problem and providing optimal controls as the planned trajectory. However, this type of optimal control method is not suitable for planning problems that need to consider obstacle avoidance.

Our method is also an optimization-based method. However, we solve the problem in lower dimensional subspaces. Therefore, the computational complexity required to solve the gradient information is lower than planning in high-dimensional space.

C. PLANNING WITH LOW-DIMENSIONAL STRUCTURE

To achieve high-dimensional motion planning, in addition to generic heuristic-based strategies for reducing the decision space, another intuitive way to bias the sampling or optimization direction is to conduct planning with low-dimensional structures. The high-dimensional planning method in [16] applies a low-dimensional structure to bias the sampling direction in the high-dimensional configuration space. The low-dimensional structure can be defined manually or determined automatically from the problem data. For certain robots, such as manipulators or mobile manipulators, the end-effector workspace and the base movement space are obvious low-dimensional subspaces [17], [18].

Instead of using the low-dimensional structure as a heuristic to bias the sampling in high-dimensional space, the planning can be done directly in the low-dimensional subspace, and then, the trajectory in the lower dimensional space is mapped to the original space via some constraints on the subspace (null space [1]) complementary to the low-dimensional subspaces, such as energy minimization. The task-space RRT proposed in [19] grows the tree in the low-dimensional task space rather than in the full configuration space. The task space denotes the workspace of the end effector. Then, the high-dimensional trajectory is mapped from the low-dimensional trajectory using the inverse kinematics model of the robot. Another example is the decompositional planning of legged robot locomotion [20]. It is relatively easy to find a feasible or optimal low-dimensional trajectory for the center of mass of the robot. Then, using this trajectory as a path to be followed by the robot, the high-dimensional footstep and joint plan can be solved by search- or optimization-based methods. As opposed to planning in a hierarchical manner, Kalin et al. proposed an adaptive dimensionality planning method for high-dimensional mobile manipulator trajectory planning. Although it utilizes the low dimensionality of the end-effector workspace, similar to the approach in [19], it constructs a single tree consisting of low- and high-dimensional states.

All the low-dimensional structures mentioned above are user defined and selected manually. A few works have also attempted to automatically learn and identify the low-

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

dimensional structure of a given problem. The approach proposed in [21] used PCA to learn and identify the subspace that contains the optimal solution. Paul et al. [4] proposed a low-dimensional structure learning method by utilizing the gradient information of the cost function and used a PCA-like method to find the basis of a subspace that contains the start and goal states. Instead of learning the low-dimensional structure from the cost function, Jung-Su et al. [5] proposed a method for learning a low-dimensional latent model of a given high-dimensional robot from experts' demonstration data. The robot dynamic model is modeled by a Gaussian process dynamical model. The authors in [22] also proposed using an autoencoding network to learn the low-dimensional latent dynamic model and then searching an optimal trajectory in the latent space, ultimately decoding the low-dimensional trajectory to one in the full state space using the trained decoder. Although the lower dimensional dynamical representation learning is a promising solution for achieving the optimal control of complex high-dimensional robots, it is inefficient for planning because the training process is lengthy and requires large amounts of training data.

In this work, we use a low-dimensional structure learning method similar to that in [4]. The learned structure is directly related to the planning problem rather than simply related to the low-dimensional structure of the robot. In [4], [5], dimensionality reduction were single-structure reduction. However, in heterogeneous environments such as the scenarios in Section V, a single-structure dimensionality reduction process may fail or the learned low-dimensional structure may be only locally effective. The results in [5] also showed that the dimensionality of the latent space should be changed based on the planning problem data (different obstacle distributions). In our approach, we propose a DP-based clustering method to find heterogeneous low-dimensional structures for a given planning problem.

III. PROBLEM DEFINITION

The optimal planning problem this work attempts to address is defined as follows:

Definition 3.1 (Optimal motion planning):

$$\begin{aligned} \mathcal{U}\{\mathbf{x}\} &:= \int_0^1 (C(\mathbf{x}_t)\|\dot{\mathbf{x}}_t\|)dt \\ \mathbf{x}^*(\tau) &:= \arg\min_{\mathbf{x}} \ \mathcal{U}\{\mathbf{x}\} \\ st.\ \mathbf{x}_0^* &= \mathbf{x}_{start},\ \mathbf{x}_1^* = \mathbf{x}_{goal}. \end{aligned} \tag{1}$$

where $\mathbf{x}^*(\tau) : [0,1] \rightarrow \mathcal{C} \subset \mathbb{R}^N$ is the optimal trajectory function mapping time to the configuration of the N DOFs robot under a given objective function $\mathcal{U}\{\mathbf{x}\}$, and $\mathbf{x}_t := \mathbf{x}(\tau) \in \mathbb{R}^N$ denotes the configuration of the robot at time t. The objective function is mainly determined by the cost function $C : \mathbb{R}^N \rightarrow \mathbb{R}$. The optimized trajectory should satisfy the start $\mathbf{x}_{start}$ and goal $\mathbf{x}_{goal}$ constraints. The integrated value with respect to $C(\mathbf{x}_t)$ in the objective function relates only to the geometric shape of the trajectory $\mathbf{x}(\tau)$ in its configuration space and not to its certain time parameterization [23].

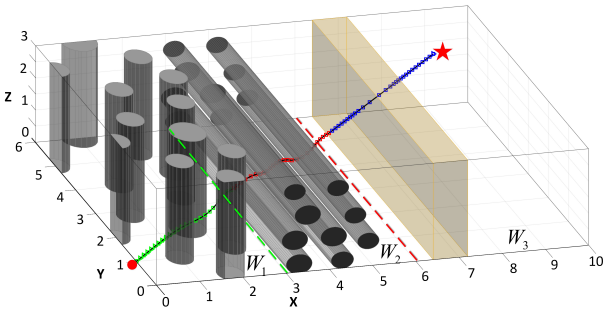

FIGURE 1: Planning in a cluttered 3D environment with dimensionality reduction.

In this paper, we propose an efficient motion planning approach that exploits the heterogeneous low-dimensional structures of the given planning problem. The next section introduces how we define and learn the heterogeneous low-dimensional structures; then. we reformulate and solve the planning problem defined in Eq. (1).

IV. METHODOLOGY

In this section, we show the technical details of our approach. To clearly describe each module of the planning framework, we use a planning task in a three-dimensional space as an example.

The planning problem in Fig. 1 involves finding an optimal trajectory with respect to a defined cost-to-go function. The heterogeneous environment in Fig. 1 includes nontraversable obstacles (the gray objects) region and a traversable high-cost region illustrated as a vertical transparent block (e.g., a high-temperature area). Assuming the configuration of the robot at time t is $\mathbf{x}_t = [x_t; y_t; z_t] \in \mathbb{R}^3$ and the main cost $C(\mathbf{x}_t)$ of our trajectory optimization problem is the collision cost. The traversable high-cost region is also treated as obstacle area when calculating the cost of a state. The obstacle cost is determined by the distance between the robot and the closest collision point.

Single-structure dimensionality reduction involves sampling the entire configuration space and learning a single low-dimensional structure. However, as shown in Section V-A, single-structure dimensionality reduction might fail when using data sampled from the entire configuration space. While no low-dimensional structure is defined for the problem in the entire configuration space, we propose that planning with low-dimensional structure can be utilized locally. First, the configuration space is decomposed into multiple sub-configuration spaces, and then the low-dimensional structures can be learned in each sub-configuration space. For example, considering the distribution of obstacles, one possible decomposition of the configuration space in Fig. 1 is to divide the space into three sub-configuration spaces denoted by $\mathcal{C}_{s-CS1}, \mathcal{C}_{s-CS2}, \mathcal{C}_{s-CS3}$. In $\mathcal{C}_{s-CS1}$ where $\{x_t \leq 3\}$, it is obvious that the decision space can be reduced to a lower dimensional subspace $\mathcal{C}_{sp1} \subset \mathbb{R}^2$. This is because all the

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

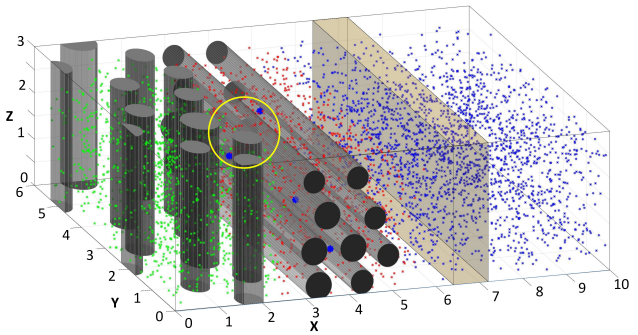

FIGURE 2: Clustered result based on DP. The color of each point denotes the space label of the point. The clustered result is not absolutely reasonable, such as the two points in the yellow circle area, but this problem can be solved by the following SVM classifier.

obstacles are vertical cylinders in $\mathcal{C}_{s-CS1}$. Then, collision avoidance along the $z-axis$ can be ignored. To describe the low-dimensional structure, a 3×2 *dimensionality reduction matrix* $\mathbf{W}_1$ can be defined and the column vectors of $\mathbf{W}_1$ are the basis of the low-dimensional subspace $\mathcal{C}_{sp1}$, resulting in $C(\mathbf{x}_t) = C(\mathbf{W}_1 {\mathbf{W}_1}^T \mathbf{x}_t) = C(\mathbf{W}_1 \mathbf{z}_t)$. For the 3D planning problem in sub-configuration space $\mathcal{C}_{s-CS1}$, the reduction matrix $\mathbf{W}_1$ can be defined simply as $\mathbf{W}_1 = [1\ 0; 0\ 1; 0\ 0]$. Details about $\mathbf{W}_1$ and $\mathbf{W}_2$ for $\mathcal{C}_{sp2}$ and $\mathcal{C}_{sp3}$ can be found in Section V-A.

The idea underlying our approach is to decompose the configuration space and to determine the dimensionality reduction matrices W_i automatically. Then, trajectory optimization is performed on those subspaces. Section IV-A presents the heterogeneous low-dimensional structures learning and Section IV-B provides our unified multiple low-dimensional subspaces trajectory optimization method.

A. HETEROGENEOUS LOW-DIMENSIONAL STRUCTURES LEARNING

In this subsection, we first present the definition of low-dimensional structure and how to learn it. Then, Section IV-A2 introduces the DP-based search of multiple low-dimensional subspaces, and Section IV-A3 presents the SVM-based subspace segmentation. Finally, the heterogeneous low-dimensional structures learning algorithm flow is presented in Section IV-A4.

1) Low-dimensional structure learning

As opposed to manually constructing the reduction matrix W_i, in this work, the lower dimensional structure for a given planning problem is trained from the problem data, such as the robot kinematics model, the cost function definition, and heterogeneous environment information. In our approach, low-dimensional structure exploiting results in multiple low-dimensional structures. For trajectory optimization in multiple low-dimensional subspaces, the label of subspace to which a low-dimensional subspace point belongs should be identifiable. To identify the label of subspace to which a low-dimensional subspace point belongs, we propose a low-dimensional structure learning method that retains some dimensions of the original configuration space. Suppose $\mathbf{x}_t = [\mathbf{p}_t, \boldsymbol{\epsilon}_t]$, the retained dimensions $\mathbf{p}_t$ is used for subspace identification of the points in low-dimensional subspaces as shown in Section IV-A3.

In this subsection, we give the concept of the low-dimensional structure and how to solve it from the problem data.

As shown in the 3D case in Fig. 1, when $\{x_t \leq 3\}$, the gradient of the cost function is $\partial C(\mathbf{x}_t)/\partial z_t = 0$. This means that the value of the cost function does not depend on the dimension z_t. Therefore, for an N-dimensional space $\mathbf{x}_t = [x_{1t}; \ldots; x_{Nt}]$ planning problem, if m dimensions exist in which the cost function does not depend on ($\partial C(\mathbf{x}_t)/\partial x_{it} = 0, i = [1, \ldots m]$), we say that the cost function has a low-dimensional structure whose dimensionality is $d = N - m$.

For a more general case, we assume there exists a rotation matrix $\mathbf{R}$ and $C(\mathbf{x}_t) = C(\mathbf{R}\mathbf{z}_t), \mathbf{z}_t = \mathbf{R}^T \mathbf{x}_t = [z_{1t}; \ldots; z_{Nt}]$. If a low-dimensional structure with dimensionality d exists, the dimensionality reduction task involves solving for a rotation $\mathbf{R}$ that makes $\partial C/\partial z_i \neq 0, i \leq d$ and $\partial C/\partial z_i = 0, d < i \leq N$ find the dimensions that make $\partial C/\partial z_i \neq 0$. Actually, the partial derivative may not exactly equal to zero. Therefore, similar to PCA, our goal is to find the first d dimensions ($d \leq N$) that have a large influence on the cost function value. Here, d is selected based on the confidence level we set [24]. Specifically, $\partial C/\partial z_i, i \leq d$ is larger than $\partial C/\partial z_i, d < i \leq N$.

Suppose that $\boldsymbol{r}_i$ is the $i-th$ column of the rotation matrix $\mathbf{R}$, and z_i is one dimension of $\mathbf{z}$:

$$\partial C/\partial z_i = \nabla C^T \boldsymbol{r}_i.$$

To obtain a large partial derivative value, $\boldsymbol{r}_i$ can be solved by the following optimization problem:

$$r_i = \arg\max_{\|\boldsymbol{r}\|=1} \mathbb{E}_x \boldsymbol{r}^T \nabla C(\mathbf{x}) \nabla C(\mathbf{x})^T \boldsymbol{r}. \tag{2}$$

We denote $\mathbf{A} := \mathbb{E}_x \nabla C(\mathbf{x}) \nabla C(\mathbf{x})^T$, and the Lagrangian function of the optimization problem in Eq. (2) as

$$\mathcal{L} = \boldsymbol{r}^T \mathbf{A} \boldsymbol{r} - \lambda(\boldsymbol{r}^T \boldsymbol{r} - 1). \tag{3}$$

The extreme value of this function can be solved by setting the derivative to zero such that

$$\begin{aligned} &\tfrac{\partial \mathcal{L}}{\partial \boldsymbol{r}} = 2\boldsymbol{A}\boldsymbol{r} - 2\lambda \boldsymbol{r} \overset{\text{set}}{=} \mathbf{0} \\ &\Longrightarrow 2\boldsymbol{A}\boldsymbol{r} = 2\lambda \boldsymbol{r} \Longrightarrow \boldsymbol{A}\boldsymbol{r} = \lambda \boldsymbol{r} \end{aligned}. \tag{4}$$

The eigenvector with the largest eigenvalue is the solution of the optimization problem defined in Eq. (2). Sorting the eigenvalues in descending order, the corresponding eigenvectors constitute the rotation matrix $\mathbf{R}$.

We select the first d columns of $\mathbf{R}$, resulting in an $N \times d$ *dimensionality reduction* matrix $\mathbf{W}$. The cost function can be approximated by a definition in d-dimensional space, $C(\mathbf{W}\mathbf{z}) \approx C(\mathbf{x}) = C(\mathbf{R}\mathbf{R}^T\mathbf{x})$. Then, the cost function

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

can be rewritten in the d-dimensional space. The basis of this lower dimensional subspace is $\mathbf{W}$.

As pointed out earlier, some dimensions of the original full-dimensional space should be retained. For example, in the 3D planning case, we assume that the subspace can be identified by the value in $x-axis$. In the I-AUV planning, we suppose that the $xyz-axes$ should be retained, which means that the position of the base must be included in the low-dimensional space. Therefore, we can find a rotation matrix $\mathbf{R}$ and update the dimensionality reduction matrix $\mathbf{W} \leftarrow \mathbf{WR}$. In the I-AUV case, the structure of the updated $\mathbf{W}$ is

$$\mathbf{W} = \begin{bmatrix} \mathbf{I} & \mathbf{0} \\ \mathbf{W}_{21} & \mathbf{W}_{22} \end{bmatrix} \tag{5}$$

where $\mathbf{I}$ is a 3×3 identity matrix.

2) Multiple low-dimensional subspace search

In the previous subsection, we described how to obtain the reduction matrix given the cost function and the configuration space. Given the configuration space shown in Fig. 1, we can sample the entire space and solve one reduction matrix [4]. However, in Fig. 1, it is obvious that the nontraversable obstacle region and traversable high-cost region in a heterogeneous environment are not uniformly distributed. Therefore it is unreasonable to reduce the dimensionality of the entire configuration space to a single lower dimensional subspace. Instead, we can divide the configuration space into multiple regions based on their lower-dimensional structures. In this section, we assume that N points were sampled in configuration space, the corresponding cost gradients $\Delta C_i, i = 1, \ldots, N$ were obtained, and that task is to cluster those points and then calculate multiple low-dimensional structures and subspaces.

However, no prior knowledge exists concerning how many subspaces should be selected to complete the planning task. Given the sampled configuration space points and their gradient, we next introduce how to divide those points into different clusters based on the DP mixture (DPM) model and the Gibbs sampling to infer the model parameters.

a: Sampled Data Modeling based on Dirichlet Process Gaussian Mixture Model

Considering the computational complexity, the Guassian mixture model (GMM) seems to be a reasonable choice to model the data. Suppose that there are K components; then, the GMM can be written as follows:

$$p\left(\check{\mathbf{x}}|\theta_1, \ldots, \theta_K\right) = \sum_{j=1}^{K} \pi_j \mathcal{N}\left(\check{\mathbf{x}}|\boldsymbol{\mu}_j, S_j\right), \tag{6}$$

where $\theta_j = \{\boldsymbol{\mu}_j, S_j, \pi_j\}$ are the parameters of the jth component, including the mean, covariance (S_j the inverse covariance matrix) and the mixing proportion π_j ($\sum_{j=1}^{K} \pi_j = 1$).

Then, the task is to find the best parameters based on the data $\check{\mathcal{X}} = \{\check{\mathbf{x}}_i\}_{i=1}^{n}$ that we obtained. In this work, we assume the ith data point is $\check{\mathbf{x}}_i = [\mathbf{p}_i, \Delta C_i]$, where $\mathbf{p}_i$ is the retained dimensions as described in Section IV-A1. For example, in the 3D case, the ith data point is defined as $\check{\mathbf{x}}_i = [\mathbf{p}_i, \Delta C_i] = [x, \delta C/\delta x, \delta C/\delta y, \delta C/\delta z]$. We propose the utilization of the DPM to model the data and automatically determine parameters θ_j.

The DP is parameterized by a base distribution G_0 and a positive scaling parameter α. Generally, given a DP prior, the DPM model can be constructed as a limit of a parametric mixture model. Given unlabeled data $\check{\mathcal{X}} = \{\check{\mathbf{x}}_i\}_{i=1}^{n}$ and considering a Gaussian mixture model for the data, the DP model can be defined as in [25]:

$$\begin{aligned} &\boldsymbol{\pi}|\alpha \sim \mathrm{Dir}(\alpha/K, \ldots, \alpha/K) \\ &l_i|\boldsymbol{\pi} \sim \mathrm{Discrete}\left(\pi_1, \ldots, \pi_K\right) \\ &\boldsymbol{\theta}_k \sim G_0(\beta) \\ &\check{\mathbf{x}}_i \,\Big|\, z_i, \{\boldsymbol{\theta}_k\}_{k=1}^{K} \sim p\left(\check{\mathbf{x}}_i|\boldsymbol{\theta}_{z_i}\right) \end{aligned} \tag{7}$$

In this DP model, given the multinomial distribution parameter $\boldsymbol{\pi}$ and the distribution parameter $\boldsymbol{\theta}_k$ of cluster k, it is assumed that each state $\mathbf{x}_i$ is sampled from one of the K clusters. $l_i \in \{1, \ldots, K\}$ is the indicator that denotes the cluster to which datum $\mathbf{x}_i$ belongs. The hyperparameter α is a symmetric Dirichlet prior for the mixing proportion $\boldsymbol{pi}$. The cluster parameters $\{\boldsymbol{\theta}_k\}_{k=1}^{K}$ are defined on a joint prior distribution $G_0(\beta)$.

By fixing all but one indicator l_i, we can obtain the conditional probability for each individual indicator using

$$p\left(l_i = k|\mathbf{l}_{-i}, \alpha\right) = \int_{\pi} p\left(l_i|\boldsymbol{\pi}\right) p(\boldsymbol{\pi}|\alpha) = \frac{n_{-i,k} + \alpha/K}{n - 1 + \alpha}, \tag{8}$$

where the subscript $-i$ indicates all the indices except for i, and $n_{-i,K}$ is the number of data points, excluding $\mathbf{x}_i$, that are associated with class k. Let $K \to \infty$, then

$$p\left(l_i|\mathbf{l}_{-i}, \alpha\right) = \begin{cases} p\left(l_i = k|\mathbf{l}_{-i}, \alpha\right) = \frac{n_{-i,k}}{n-1+\alpha} \\ p\left(l_i \neq l_{i'} \text{ for all } i \neq i'|\mathbf{l}_{-i}, \alpha\right) = \frac{\alpha}{n-1+\alpha} \end{cases}. \tag{9}$$

Based on the conditional distribution, the cluster indicator of $\mathbf{x}_i$ could be an existing component k or a new cluster.

b: Gibbs Sampling for Model Parameter Estimation

To infer the parameters in Eq. (7), we utilize the Markov chain Monte Carlo method based on Gibbs sampling. Given the data $\check{\mathcal{X}} = \{\check{\mathbf{x}}_i\}_{i=1}^{n}$ $\left(\check{\mathbf{x}}_i \in \mathbb{R}^d\right)$ and their cluster indicators $\mathcal{L} = \{l_i\}_{i=1}^{n}$, the Gibbs sampling involves iterations that alternately draw from a conditional probability while keeping other variables fixed. Recall that for each indicator variable l_i, we can derive its conditional posterior in the DPM model as follows:

$$\begin{aligned} &p\left(l_i = k|\mathbf{l}_{-\mathbf{i}}, \mathbf{x}_i, \{\boldsymbol{\theta}_k\}_{k=1}^{K}, \alpha, \beta\right) \\ =&p\left(l_i = k|\mathbf{x}_i, \mathbf{l}_{-\mathbf{i}}, \{\boldsymbol{\theta}_k\}_{k=1}^{K}\right) \\ \propto&p\left(l_i = k|\mathbf{l}_{-\mathbf{i}}, \{\boldsymbol{\theta}_k\}_{k=1}^{K}\right) p\left(\check{\mathbf{x}}_i|l_i = k, \{\boldsymbol{\theta}_k\}_{k=1}^{K}\right) \\ =&p\left(l_i = k|\mathbf{l}_{-\mathbf{i}}, \alpha\right) p\left(\check{\mathbf{x}}_i|\boldsymbol{\theta}_k\right) \end{aligned} \tag{10}$$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

$$p\left(\boldsymbol{\theta}_k|\check{\mathbf{x}}_k,\beta\right)=p\left(\boldsymbol{\theta}_k|\beta\right)\prod_{i=1}^{|\check{\mathbf{x}}_\mathbf{k}|}p\left(\check{\mathbf{x}}_{k_i}|\boldsymbol{\theta}_k\right) \tag{11}$$

where $p\left(l_i=k|\mathbf{l}_{-\mathbf{i}},\alpha\right)$ is solved by Eq. (8), and $p(\check{\mathbf{x}}_k|\boldsymbol{\theta}_{l_k})$ is the likelihood for the current state $\mathbf{x}_i$. The distribution parameters $\boldsymbol{\theta}_{l_k}$ can be determined by maximizing the conditional posterior. If the current set for cluster k is $\check{\mathbf{x}}_k$ and there are $n_k=|\check{\mathbf{x}}_k|$ elements in this set, then based on the prior $G_0(\beta)$, the DPM model learns $\boldsymbol{\theta}_{l_k}$ by maximizing the posterior:

The update process is summarized by the following steps:

- update the mixture Gaussian model parameters $(\boldsymbol{\mu}_j, S_j)$ as in Eq. (11) and the hyperparameters;
- update the indicators of each datum $\mathbf{x}_i$ based on Eq. (10)
- update the concentration parameter α of the DPM model.

In this work, we applied the update rule detailed in [26]. In our model, the data point $\check{\mathbf{x}}_i=[\mathbf{p}_i,\Delta C_i]$ for clustering consist of the retained dimensions and the cost gradient of the sampled point $\mathbf{x}_i$.

Algorithm 1: DP-based multiple low-dimensional structures learning

Input: Start $\mathbf{x}_s$ and goal $\mathbf{x}_g$; cost function, $C(\mathbf{x})$; map $\mathcal{M}$;
Output: Multiple low-dimensional structures $\{\mathbf{W}_1,\mathbf{W}_2,\ldots,\mathbf{W}_E\}$;
Step1. DP-based subspaces search:
while *not succeed or t<maximun time limit* **do**
  1:$\{\mathbf{x}_i,\ i=1,\ldots,N_{sample}\}$ ←Randomly sampled configurations
  2: $\{\Delta C(\mathbf{x}_i)\}$ ←Cost gradients at sampled points
  3: Initialize number of clusters k of the DPM model
  /*Run clustering method based on DP and output cluster low-dimensional space*/
  4: $Label(\mathbf{x})\in\{1,\ldots,E\}\leftarrow DP(\mathbf{x},\Delta C(\mathbf{x}))$;
end
Step 2. SVM-based subspace segmentation:
/*SVM training process*/
5: $\mathcal{M}\leftarrow SVM(\mathbf{x},Label(\mathbf{x}_{1,\ldots,N_{sample}}))$;
/*SVM predicting process*/
6: New labels of all points $Label(\mathbf{x})\leftarrow\mathcal{M}(\mathbf{x})$
Step 3. Multiple low-dimensional structures learning:
/*Assuming the space label is $l\leq E$*/
for $l\leq E$ **do**
  7: Find all sampled data in space l;
  8: $\mathbf{W}_l\leftarrow$ Low dimensional structure learning based on $\Delta C(\mathbf{x}_l), Label(\mathbf{x}_l)$
end
Return $\{\mathbf{W}_1,\mathbf{W}_2,\ldots,\mathbf{W}_E\}$;

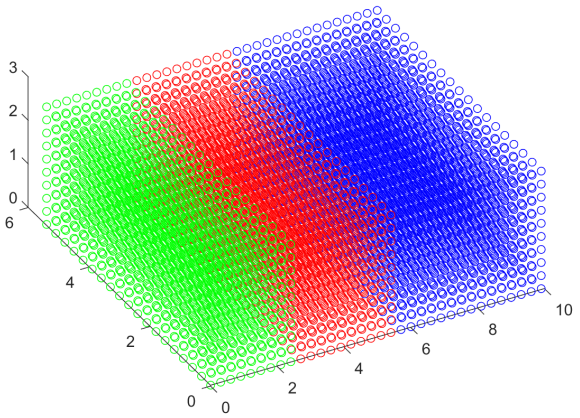

FIGURE 3: The SVM classification result. Given the position of each sampled point in the 3D space, SVM outputs the space label of the point denoted by different colors.

3) Subspace segmentation based on the clustered sampled points and SVM

Based on the clustered sampled data points, a classifier can be applied to segment the configuration space, resulting in multiple low-dimensional subspaces with dimensionality d_i ($d_i\leq n$).

In this paper, we utilize the popular classifier SVM for nonlinear classification. The Gaussian radial basis function (RBF) is used as the kernal function of the SVM model.

$$k(\mathbf{x}_i,\mathbf{x}_j)=e^{-\gamma\|\mathbf{x}_i-\mathbf{x}_j\|^2},\gamma>0. \tag{12}$$

The training data are the sampled points and the cluster labels of those points. The classifier outputs a decision function $\mathcal{M}$. Given any state to $\mathcal{M}$, a label of the state will be obtained that will be used in the multiple-subspace trajectory optimization.

4) Heterogeneous low-dimensional structures algorithm overview

Now, based on the low-dimensional structure definition and learning method in Section IV-A1, the multiple subspaces search in Section IV-A2, and the SVM-based subspace segmentation, the DP-based multiple low-dimensional structures learning algorithm in our work is summarized in Algorithm 1. Low-dimensional structure learning in Section IV-A1 is based on the data in a single subspace. Therefore, the first two steps in Algorithm 1 are: sampling and clustering, and SVM-based segmentation. The last step is the low-dimensional structures learning based on the clustered data in each subspace. This algorithm finally returns multiple low-dimensional structures $\{\mathbf{W}_1,\mathbf{W}_2,\ldots,\mathbf{W}_E\}$.

B. UNIFIED TRAJECTORY OPTIMIZATION WITH HETEROGENEOUS LOW-DIMENSIONAL STRUCTURES

Planning using the single-structure dimensionality reduction method proposed in [27] generates the optimal trajectory by

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

optimization over only one low-dimensional subspace that contains the start and goal states. The defined optimization problem can be solved via any basic optimization method such as dynamic programming or gradient-based optimization. For planning with single-structure methods, the dimensions of the optimization variables remains unchanged during optimization.

However, in our case, the trajectory is optimized in multiple low-dimensional subspaces; hence, the subspace to which one waypoint belongs may change from the current subspace to a different subspace; consequently, the dimensionality of the optimization variables also will change. For example, in Fig. 4, the waypoints g_1^l, g_m^l are called switching points (the subspace to which the next waypoint belongs is different from the subspace to which the switching point belongs). After one iteration of the trajectory optimization, the subspace of switching point g_1^l belongs may change from $\mathcal{C}_{sp1}$ to $\mathcal{C}_{sp2}$. Therefore, dynamic programming or gradient-based optimization can not be directly applied. In our solution, we divide the optimization process into two steps. In the first step, optimization is performed in the low-dimensional subspaces. In this step, trajectory optimization considers the transversality condition of the switching points between two subspaces, but it does not consider the transversality in the original full-dimensional space. The second step of the trajectory optimization considers the constraints in the original full-dimensional space and maps the optimized low-dimensional trajectory to full-dimensional space.

1) Trajectory planning in low-dimensional subspaces

Based on the planning problem in *Definition 3.1*, we first define the trajectory optimization problem in multiple low-dimensional subspaces, and then give our gradient-based optimization approach and the transversality condition in low-dimensional trajectory optimization.

a: Low-dimensional trajectory planning problem definition

Many parameterization methods exist for representing a trajectory, such as piecewise polynomial trajectories and B-spline trajectories. We use a uniform discretization that samples the trajectory function over equal time steps of length Δt : $\xi_{\mathbf{x}} \approx \left(\mathbf{x}_1^{\mathrm{T}}, \mathbf{x}_2^{\mathrm{T}}, \ldots, \mathbf{x}_i^{\mathrm{T}}, \mathbf{x}_{i+1}^{\mathrm{T}}, \ldots, \mathbf{x}_n^{\mathrm{T}}\right)^{\mathrm{T}} \in \mathbb{R}^{n \times N}$, where $\mathbf{x}_0$ and $\mathbf{x}_{n+1}$ are the fixed starting and goal point of the trajectory, respectively. Suppose there are E low-dimensional subspaces for a given planning problem, and the subspace index of the ith waypoint is denoted by $l_i \in \mathcal{E} = \{1, \ldots, E\}$. The basis of the subspace where the ith waypoint located is denoted by $\mathbf{W}_i \in \{\mathbf{W}_1, \ldots, \mathbf{W}_E\}$.

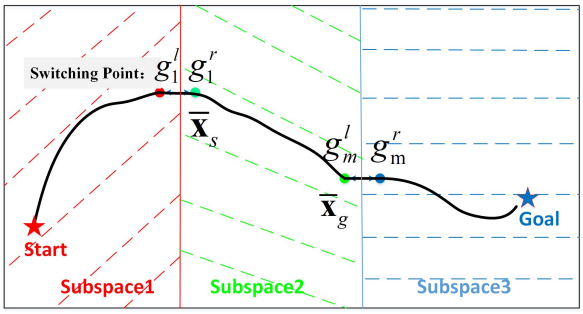

FIGURE 4: Planning in multiple low-dimensional subspaces. Considering the subspaces switching and the smoothness of trajectory at switching points.

Definition 4.1: Planning in multiple low-dimensional subspaces :

$$\begin{aligned}\xi_{\mathbf{z}} :&= \arg\min_{\xi_{\mathbf{z}}} \mathcal{U}(\xi) \\ &:= \arg\min_{\xi_{\mathbf{z}}} \sum_{i=1}^{n} C(\mathbf{W}_i \mathbf{z}_i(t)) \|\dot{\mathbf{z}}_i\| + \|\dot{\mathbf{x}}_{common}\|^2, \quad (13a) \\ z(0) &= \mathbf{W}_0 \mathbf{x}_s, \qquad z(n) = \mathbf{W}_n \mathbf{x}_g, \quad (13b)\end{aligned}$$

where $\xi_{\mathbf{z}} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}$ denotes the trajectory in the subspaces and $\mathbf{z}_i = \mathbf{W}_i^{\mathrm{T}} \mathbf{x}_i$ is the ith waypoint of the trajectory. The subscript $common$ in $\dot{\mathbf{x}}_{common}$ denotes the velocity of the common dimensions of the state between two adjacent points. The velocity of the trajectory in one subspace can be defined easily. However, as shown in Fig. 4, in multiple subspaces trajectory optimization, the velocity term around the switching points should be treated carefully. In our solution, we define the following velocity term, which considers the process of switching between subspaces.

$$\begin{aligned}\|\dot{\mathbf{x}}_{common}\| &= norm(\mathcal{W}(\mathbf{W}_i \mathbf{z}_i - \mathbf{W}_{i-1} \mathbf{z}_{i-1})), \\ \mathcal{W} &= \mathbf{W}_{i-1} \mathbf{W}_{i-1}^T + \mathbf{W}_i \mathbf{W}_i^T - \mathbf{I}.\end{aligned} \quad (14)$$

When the subspace label of the $(i-1)$-th waypoint is the same as that of the ith waypoint, $\mathcal{W}$ is approximate to an identity matrix. This velocity term denotes the velocity of the trajectory in a single low-dimensional subspace. At the switching points, $\mathbf{W}_{i-1}$ and $\mathbf{W}_i$ denote different subspaces bases. The velocity term attempts to reduce the displacement at the common dimensions of two states in two different subspaces by using the factor $\mathcal{W}$. The velocity term guarantees the smoothness of the low-dimensional trajectory in multiple subspaces.

b: Gradient-based optimization with transversality condition

To solve the non-convex optimization problem in Eq. (13), we utilize a gradient-based approach to iteratively optimize $\mathcal{U}(\xi_{\mathbf{z}})$. At each iteration the objective function is approximated by its first-order Taylor expansion around the trajectory $\xi_{\mathbf{z}}$:

$$\mathcal{U}[\xi_{\mathbf{z}} + \Delta\xi_{\mathbf{z}}] \approx \mathcal{U}[\xi_{\mathbf{z}}^k] + \nabla\mathcal{U}[\xi_{\mathbf{z}}]\Delta\xi_{\mathbf{z}}. \quad (15)$$

We use the Gaussian-Newton algorithm to iteratively improve the solutions. Suppose that the incremental solution at each iteration is $\Delta\xi_{\mathbf{z}}$; then, we use the following updating rule:

$$\begin{aligned}\bar{\xi}_{\mathbf{z}}^{k+1} &\leftarrow \xi_{\mathbf{z}}^{k} + \Delta\xi_{\mathbf{z}},\\ \bar{\xi}_{\mathbf{x}}^{k+1} &= \{\mathbf{W}_1^k\bar{\mathbf{z}}_1^{k+1}, \mathbf{W}_2^k\bar{\mathbf{z}}_2^{k+1}, \ldots, \mathbf{W}_n^k\bar{\mathbf{z}}_n^{k+1}\},\end{aligned} \tag{16}$$

where $\bar{\xi}_{\mathbf{z}}^{k+1}$ is an updated trajectory in multiple low-dimensional subspaces, and k denotes the number of iteration, and $\bar{\xi}_{\mathbf{x}}$ is a trajectory in the original full-dimensional space.

Then, using the trained SVM model shown in Section IV-A3 and the new waypoints trajectory $\bar{\xi}_{\mathbf{x}}^{k+1}$, we can obtain the label l of each waypoint, which denotes the label of the subspace to which each waypoint belongs. Although the trajectory $\bar{\xi}_{\mathbf{x}}^{k+1}$ is not the final full-dimensional trajectory in the original configuration space, Eq. (5) guarantees that some dimensions of the coordinates are in the original space such that the trained SVM model can be utilized to update the subspace labels of the waypoints.

$$l(\bar{\xi}_{\mathbf{x}_i}^{k+1}) \leftarrow SVM(\bar{\xi}_{\mathbf{x}_i}^{k+1}). \tag{17}$$

After the low-dimensional trajectory updating in Eq. (16) and the waypoints subspace labels updating in Eq. (17), the subspace label of a waypoint may change. Suppose that one point $\mathbf{x}_i$ in subspace l_i is transferred into a new subspace l_i'; the basis of the new subspace l_i' is $\mathbf{W}_i'$, and the first waypoint of the low-dimensional trajectory in subspace l_i' is $\mathbf{z}_i'$ (e.g., in Fig. 4, g_1^r is the first waypoint of the trajectory in $\mathcal{C}_{sp2}$). To obtain the waypoint's new low-dimensional coordinate $\mathbf{z}_i^{k+1}$ in subspace l_i', we apply the following *transversality condition* [7]:

$$\mathbf{z}_i^{k+1} \leftarrow \arg\min\|\mathbf{z}_i^{k+1} - \mathbf{z}_i'\|. \tag{18}$$

The purpose of the transversality condition is to guarantee the continuity of the trajectory in each subspace. Then, if the subspace label of a waypoint changes, we execute the following switching equation:

$$\bar{\xi}_{\mathbf{x}_i}^{i+1} = \mathbf{W}_i^k\bar{\mathbf{z}}_i^{k+1} + (\mathbf{I} - \mathbf{W}_i^k\mathbf{W}_i^{k^T})\mathbf{W}_i'\mathbf{z}_i'. \tag{19}$$

Based on the new labels $l(\bar{\xi}_{\mathbf{x}}^{k+1})$ of each waypoint and the updated trajectory $\bar{\xi}_{\mathbf{x}}^{k+1}$, we obtain the updated low-dimensional trajectory

$$\xi_{\mathbf{z}}^{i+1} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_n\}^{i+1}. \tag{20}$$

2) Full-dimensional trajectory smoothing

The above trajectory optimization is processed in the multiple lower dimensional subspaces. However, to complete the planning task, we need to obtain a trajectory in the original high-dimensional decision space. Therefore, we need to provide a mapping function λ,

$$\xi_{\mathbf{x}} \leftarrow \lambda(\xi_{\mathbf{z}}), \tag{21}$$

where $\xi_{\mathbf{x}}$ is the trajectory in the original full-dimensional space.

Transformations from the original space to subspaces $\xi_{\mathbf{z}} \leftarrow \lambda^{-1}(\xi_{\mathbf{x}})$ are many-to-one transitions. In this work, we consider a single-query planning task. Therefore, in one subspace, we only need to guarantee that the start state $\mathbf{x}_s$ and goal state $\mathbf{x}_g$ lie in the mapped trajectory. As shown in Fig. 4, when planning in multiple subspaces, the smoothness of the full-dimensional trajectory at the switching points should also be considered.

Definition 4.2 (Mapping trajectory in low-dimensional subspaces to original high-dimensional space):

$$g_k := \arg\min_{g_k} \sum_{k=1}^{m} \|g_k^l - g_k^r\|^2 + \mathcal{C}_{ns-dist}$$

$$st.\quad \mathbf{W}_1\mathbf{x}_s = z_1,\ \ \mathbf{W}_k g_k^{l,r} = z_k^{l,r},\ \ \mathbf{W}_K\mathbf{x}_g = z_n, \tag{22a}$$

$$\mathbf{x}_i = \mathbf{W}\mathbf{z}_k + (\mathbf{I} - \mathbf{W}\mathbf{W}^{\mathrm{T}})\bar{\mathbf{x}}_s + (\mathbf{I} - \mathbf{W}\mathbf{W}^{\mathrm{T}})ys, \tag{22b}$$

$$y = \bar{\mathbf{x}}_g - \bar{\mathbf{x}}_s, \tag{22c}$$

$$s = \frac{\|\mathbf{z}_i - \bar{\mathbf{z}}_s\|}{\|\mathbf{z}_i - \bar{\mathbf{z}}_g\| + \|\mathbf{z}_i - \bar{\mathbf{z}}_g\|}, \tag{22d}$$

where the objective function consists of two terms: the first term is transversality condition in full-dimensional space optimization, which guarantees the smoothness of trajectory at switching points, and $\mathcal{C}_{ns-dist}$ is the length of the trajectory in the null space of the low-dimensional trajectory. In Eq. (22a), g_k^l and g_k^r are two points at the switching area as shown in Fig. 4, Eq. (22b) is the coordinate transformation from the low-dimensional subspace to the original space and $\bar{\mathbf{x}}_s$, $\bar{\mathbf{x}}_g$ are the start and end states of the trajectory in one subspace.

The nonlinear optimization problem defined in Eq. (22) can easily be solved using SQP [28].

C. COMPUTATIONAL COMPLEXITY ANALYSIS

In our framework, the computational complexity stems mainly from two components: the basis learning of the lower dimensional subspaces and the gradient-based trajectory optimization. We note that for the first part, although large numbers of points should be sampled to guarantee the consistency of the learned dimensionality reduction matrices, the cost gradient calculation could be processed in parallel. It is relatively easy to implement parallel computing on computers with multiple cores. Suppose that the original high-dimensional space dimensionality is N, for each iteration, the computational complexity for the gradient-based optimization would be $\mathcal{O}(kN^3)$, where k is the number of waypoints in the trajectory. For single-structure dimensionality reduction, assume the dimensionality of the low-dimensional subspace is d_s; then, the computational complexity would be $\mathcal{O}(kd_s^3)$. In our approach, we assume that the dimensionality of the subspace to which the ith waypoint belongs is $d_{l(i)}$ ($d_{l(i)} \leq d_s$, where $l(i)$ is the subspace indicator of the waypoint as shown in Eq. (17)) . Then, the computational complexity of our approach is $\sum_{1\leq i\leq k}\mathcal{O}(d_{l_i}^3)$. Compared with single-structure dimensionality reduction method [2],

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

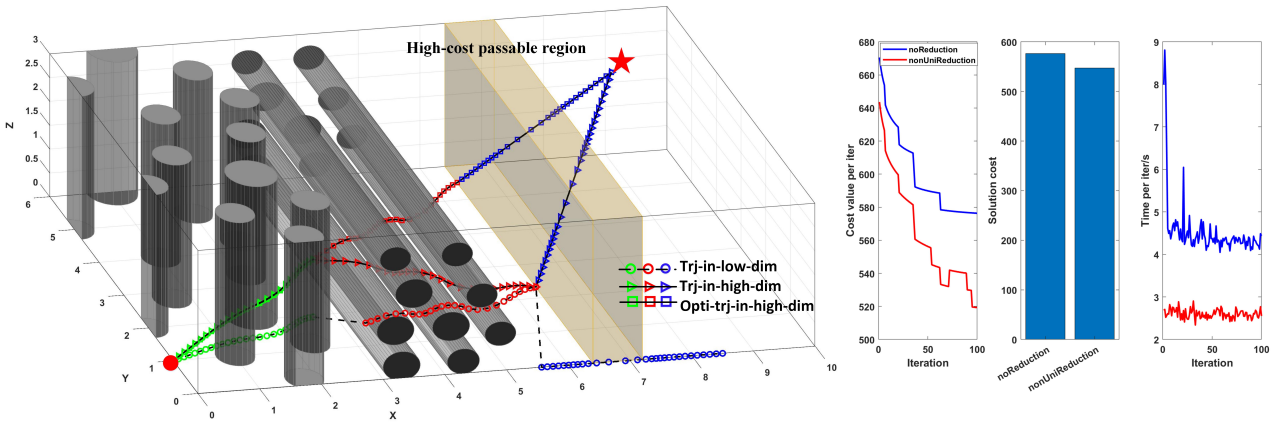

FIGURE 5: Planning in multiple low-dimensional subspaces with bases $\mathbf{W}_{i,\ldots,j}$. The red dot and pentagram denote the start $\mathbf{x} = [0; 1; 0]$ and goal $\mathbf{x} = [8.5; 4.5; 2.8]$ points, respectively. The colors of the waypoints of the trajectories denote the subspace to which each waypoint belongs. In regions with cluttered obstacles, the primary task is to guarantee the clearance of the trajectory. To reduce the cost of the trajectory in the traversable high-cost region, the robot should move quickly.

our approach reduces the computational complexity to varying degrees based on the complexity of the problem data.

V. SIMULATIONS AND RESULTS

The solution presented in Section IV is summarized in Algorithm 2. In this section, we first use 3D space planning to verify our algorithm clearly; then, the results of a high-DOF I-AUV planning simulation are presented.

Algorithm 2: Planning with heterogeneous dimensionality reduction

Input: Start $\mathbf{x}_s$ and goal $\mathbf{x}_g$; cost and objective function, $C(\mathbf{x})$ and $\mathcal{U}(\xi)$; map $\mathcal{M}$;
Output: The optimal trajectory in Eq. (21);
1. $\{\mathbf{W}_1, \mathbf{W}_2, \ldots, \mathbf{W}_E\} \leftarrow$ Low-dimensional structures from Algorithm 1;
2. Planning in subspaces as in Eq. (13);
3. Mapping $\xi_{\mathbf{z}}$ to the original space trajectory $\xi_{\mathbf{x}}$ as in Eq. (22)
return $\xi_{\mathbf{x}}$;

A. THREE-DIMENSIONAL SPACE PLANNING

Continue the 3D space planning example in previous sections. As shown in Fig. 5, we assume that there are non-traversable obstacles (the gray objects) and a traversable high-cost region illustrated as a vertical transparent block (e.g., a high-temperature area), assumed to extend ad infinitum in directions orthogonal to the vertical direction. In the cost function definition of this problem, we assume that the cost introduced from the traversable region has the cost formulation as the obstacle areas.

We select an exponential function as the obstacle cost function. Given a position $\mathbf{x}$ of the robot in a signed distance map, the cost function $C(\mathbf{x})$ is

$$C(\mathbf{x}) = \bar{C}(\delta(\mathbf{x})) = \alpha \cdot \exp\left(-\left(\delta - \delta_0\right)/r\right), \qquad (23)$$

where δ is the distance between the robot and the closest obstacle point, δ_0 is the threshold where the cost starts to rapidly rise, and r controls the rate of the function's rise.

The random state sampling and low-dimensional subspace exploitation results are shown in Fig. 2. Figure 3 shows that the map is divided into three parts by the SVM classification. According to Eq. (5), the reduction matrix in this example is quite simple. The map is divided into three parts, and the dimensionality reduction matrices are as follows:

$$\mathbf{W}_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}^T, \mathbf{W}_2 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}^T, \mathbf{W}_3 = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}^T.$$

In Fig. 5, Trj-in-low-dim is the optimized trajectory in the low-dimensional subspaces. As described in Section IV-B, the smoothness of this trajectory in the original full-dimensional space is not guaranteed. There are two cost terms in Eq. (22). Trj-in-high-dim is an optimized trajectory considered for the first cost term in Eq. (22), and Opti-trj-in-high-dim is the optimal trajectory.

To demonstrate the advantages of planning with heterogeneous or nonuniform dimensionality reduction (nonUniReduction), two other methods, planning without dimensionality reduction (noReduction) and planning with uniform dimensionality reduction (UniReduction), are compared with our approach. All methods use the same gradient-based trajectory optimization method as described in Section IV-B. When planning with uniform dimensionality reduction, two strategies can be utilized for the low-dimensional structure, as shown in [2] and [4], i.e., SLASHDP and LDD, respectively, which uniformly reduce the original state space to a single low-dimensional subspace. In SLASHDP, the optimal trajectory is searched in the single low-dimensional space. In LDD, the first step is to learn the low-dimensional subspace basis, and the second step selects a subset of those bases to optimize according to their relative importance. By optimizing the path in the selected subspace, the projected

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

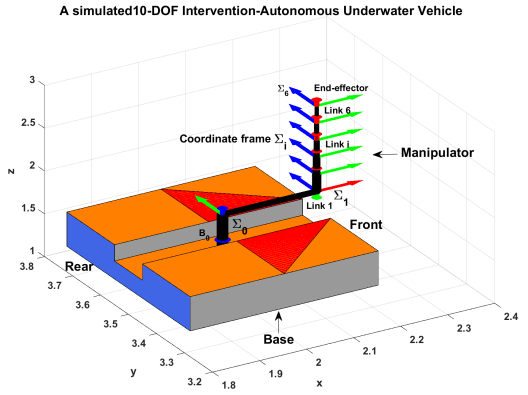

FIGURE 6: A simulated 10-DoF I-AUV with 6 links. The blue axis is the rotation axis fixed at the start point of each joint. The robot model and parameters setup are based on the I-AUV presented in [29].

path in the remainder of the space will remain fixed. This strategy improves the solutions monotonically. Intuitively, we could also use this strategy in a more efficient way: in each iteration, we could use the learned basis of each low-dimensional subspace.

In [2], given the planning problem in Fig. 5, the three eigenvalues of $\mathbf{M}$ as shown in Eq. (4) from our test are $(1.19, 0.27, 0.25)$. These three values are relatively similar; therefore, in this case, SLASHDP is inefficient. In the left-hand figure, Fig. 5, we compare the convergences of the noReduction and nonUniReduction methods, their solution costs, and their time consumption for each iteration. The solution cost is the final objective function value in the original space as defined in Eq. (1). The results show that using our method, the cost function converges to a relatively low cost and the time consumption for each iteration is lower than that of the noReduction method.

B. HIGH-DOF I-AUV PLANNING

We simulated a 10-DOF I-AUV with a holonomic vehicle base driven by thrusters and a 6-DOF manipulator as shown in Fig. 6. The model parameter setup is based on the underwater robot presented in [29]. This robot can be used to perform data collection, mobile manipulation or cleaning. We assume that the state of the I-AUV in the configuration space is $\mathbf{x} = [x, y, z, \Psi, \theta_1, \ldots, \theta_6]$. The goal is to find an optimal path $\xi_{\mathbf{x}}$ in the configuration space.

The cost function used for low-dimensional structure exploration is

$$C(\mathbf{x}) = \sum_{i=1}^{n_s} \exp\left(-\left(\delta^i - \delta_0^i\right)/r\right), \tag{24}$$

The obstacle cost function of the I-AUV is similar to the definition in Eq. (23). δ^i denotes the position of the collision checkpoint on the robots. In this work, we assume that the collision checkpoints include the points on the edges of the vehicle base and the links of the manipulator. The total number of collision check points is n_s [13].

In this experiment, for mobile manipulator planning, we consider navigation tasks rather than manipulation tasks [3]. The start position is relatively far away from the goal, as shown in Fig. 7 and the planning task involves planning a collision-free trajectory for the base and arm from the start configuration to the goal in a coupled way [1], [30].

1) Sampling strategy for high-DoF I-AUV

Cost gradient sampling described in Section IV-C is a time-consuming component. To learn the low-dimensional structure of the cost function efficiently for high-DoF robots, as opposed to sampling randomly, we can use multiple arm configurations, such as the configuration in Fig. 6 , which has a maximum collision probability (the arm is fully extended upward). In our trials, we use 100,000 cost gradient samples in parallel. For a given task, the number of samples could also be determined by the running time limits of the algorithm. Other workspace analysis method can also be applied to the bias the sampling, such as the Generalized Voronoi Graph-based (GVG) narrow passages search [30] and learning-based non-uniform sampling strategies proposed in [12]. In those works, more points are sampled at the narrow passages areas or other important areas.

Given the cost function in Eq. (24), the robot in Fig. 6, and the map in Fig. 7, two dimensionality reduction matrices or the basis of the subspaces are obtained, $\mathbf{W}_1$ and $\mathbf{W}_2$, as follows:

$$\mathbf{W}_2 = \begin{bmatrix} 1 & 0 & 0 & 0.0000 \\ 0 & 1 & 0 & 0.0000 \\ 0 & 0 & 1 & -0.0019 \\ 0 & 0 & 0 & -0.8285 \\ 0 & 0 & 0 & -0.5599 \\ 0 & 0 & 0 & -0.0044 \\ 0 & 0 & 0 & -0.0009 \\ 0 & 0 & 0 & -0.0028 \\ 0 & 0 & 0 & -0.0013 \\ 0 & 0 & 0 & -0.0010 \end{bmatrix},$$

$$\mathbf{W}_2 = \begin{bmatrix} 1 & 0 & 0 & -0.0023 & 0.0005 \\ 0 & 1 & 0 & -0.0040 & 0.0021 \\ 0 & 0 & 1 & 0.0028 & -0.0084 \\ 0 & 0 & 0 & 0.7008 & 0.0452 \\ 0 & 0 & 0 & 0.7101 & 0.0510 \\ 0 & 0 & 0 & 0.0531 & -0.7766 \\ 0 & 0 & 0 & 0.0355 & -0.5243 \\ 0 & 0 & 0 & 0.0209 & -0.3088 \\ 0 & 0 & 0 & 0.0099 & -0.1432 \\ 0 & 0 & 0 & 0.0029 & -0.0390 \end{bmatrix}.$$

The matrix $\mathbf{W}_1$ shows that in subspace 1, in addition to x, y, z, the movements of ψ, θ_1 are more important than those in other dimensions, while in subspace 2, ψ, θ_1, θ_2 are the dimensions in which the value of the objective function varies

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

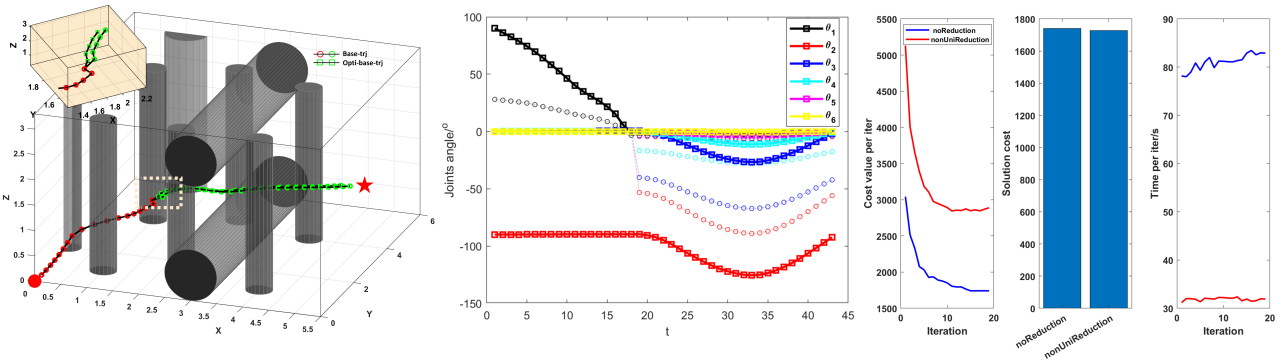

FIGURE 7: Planning in multiple low-dimensional subspaces with bases $\mathbf{W}_{i,\ldots,j}$. The red dot and the pentagram denote the start $\mathbf{x} = [0; 1; 0]$ and goal $\mathbf{x} = [8.5; 4.5; 2.8]$ points, respectively. The colors of the waypoints of the trajectories denote the subspace to which each waypoint belongs.

substantially.

2) Trajectory optimization results

The planning task involves planning a low-cost trajectory resulting in a collision-free base and arm trajectory from the initial configuration to the goal configuration. Generally, for high-dimensional space planning, gradient-descent optimization can readily become trapped in local minima, resulting in relatively high costs or infeasible trajectories. Many methods are available to alleviate this problem such as the Hamiltonian Monte Carlo algorithm used in [13]. In this work, we use the initial trajectories search method proposed in our previous work to avoid the local minima problem [1]. The searched initial solutions are in multiple homotopy classes and solved through the RRTconnect and polynomial trajectory optimization-based methods for the initial base trajectory search and a null space saturation controller for the initial manipulator path search.

The simulation results of one planning task are shown in Fig. 7. The start and goal configurations are $\mathbf{x}_{start} = [0.1; 0.1; 0.1; 0; \pi/2; -\pi/2; 0; 0; 0; 0]$; and $\mathbf{x}_{goal} = [5; 3; 1.5; 0; 0; -\pi/2; 0; 0; 0; 0]$, respectively. The optimized I-AUV base trajectory is shown in Fig. 7. The zoomed-in area shows that although the smoothness of the trajectory in each subspace is guaranteed, it should be post-processed in the connection area. The manipulator trajectory in Fig. 7 shows that in this case, to complete the task, θ_1 and θ_2 change more than other dimensions, which means that θ_1 and θ_2 are the most important dimensions.

To prove the effectiveness, we compared both the optimality of the planned trajectories and the time consumption for trajectory optimization as in the comparison in the previous 3D space planning case. We tested our algorithm using MATLAB 2017b running on a quad-core 2.8 GHz Intel i7 CPU with 16GB of RAM. As mentioned in Section IV-C, the runtime of the cost gradient sample can be reduced by using parallel computing on multicore computers. In simulations, using the parallel computing command *Parfor*, our test results show that on a computer with a k-core CPU, the nonparallel computation time is k times that of the parallel calculation. Therefore, in the subsequent comparisons, we do not consider the time for the cost gradient sample. As shown in the left figure in Fig. 7, the noReduction and nonUniReduction methods converge to local minima within 20 iterations. Their final solution costs are very similar; however, the computation time per iteration of the noReduction method is almost 2.7 times that of the nonUniReduction method.

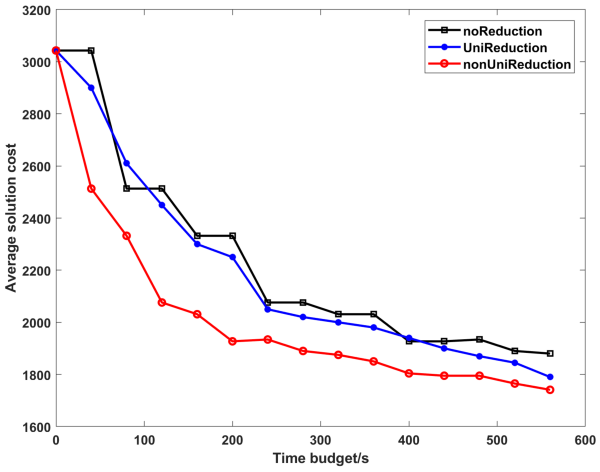

FIGURE 8: Algorithm performances under fixed time budgets.

In practice, planners usually need to consider the time budget for task execution. Therefore, to compare the efficiency of different methods, we evaluated the solution cost for a given planning time. For the planning problem in Fig. 7, we executed each method 20 times. Fig. 8 shows that the average solution costs obtained by our approach are lower than those of the other methods for the given time budgets. We should note that the comparisons described above are not absolutely comprehensive-there are no benchmarks for comparison, and code for the other methods is unavailable,

as they are not open source. Thus, the performances of those methods depend on the implementation details.

VI. CONCLUSIONS AND FUTURE WORK

In this work, we presented a novel motion planning framework for planning problems with low-dimensional structures. The proposed approach utilizes tools from data analysis (DP and SVM) to explore the heterogeneous low-dimensional structures. Numerical simulation results demonstrated the efficiency of the unified multiple low-dimensional subspaces trajectory optimization approach.

Although we present a unified optimization-based trajectory planning approach that operates in multiple subspaces, other generic trajectory planning methods could also be applied to solve multi-subspace planning problems, such as the sampling- and search-based methods [30].

The proposed approach improves the efficiency if and only if one or multiple low-dimensional structures for a given planning problem exist. In the future, we plan to develop more general dimensionality reduction methods by using neural network-based approaches to reduce the complexity in obtaining the heterogeneous low-dimensional structures, via experiences and analogy [31], [32].

REFERENCES

[1] H. Yu, W. Lu, and D. Liu, "A unified closed-loop motion planning approach for an i-auv in cluttered environment with localization uncertainty," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2019, pp. 4646–4652.

[2] P. Vernaza and D. D. Lee, "Efficient dynamic programming for high-dimensional, optimal motion planning by spectral learning of approximate value function symmetries," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2011, pp. 6121–6127.

[3] K. Gochev, A. Safonova, and M. Likhachev, "Planning with adaptive dimensionality for mobile manipulation," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2012, pp. 2944–2951.

[4] P. Vernaza and D. D. Lee, "Learning and exploiting low-dimensional structure for efficient holonomic motion planning in high-dimensional spaces," Int. J. Robot. Res., vol. 31, no. 14, pp. 1739–1760, 2012.

[5] J.-S. Ha, H.-J. Chae, and H.-L. Choi, "Approximate inference-based motion planning by learning and exploiting low-dimensional latent variable models," IEEE Robotics and Automation Letters, vol. 3, no. 4, pp. 3892–3899, 2018.

[6] J. Joseph, F. Doshi-Velez, A. S. Huang, and N. Roy, "A bayesian nonparametric approach to modeling motion patterns," Auton. Robot., vol. 31, no. 4, p. 383, 2011.

[7] W. Lu and S. Ferrari, "An approximate dynamic programming approach for model-free control of switched systems," in Proc. IEEE Conf. Decis. Control (CDC), 2013, pp. 3837–3844.

[8] B. Cohen, S. Chitta, and M. Likhachev, "Single-and dual-arm motion planning with heuristic search," Int. J. Robot. Res., vol. 33, no. 2, pp. 305–320, 2014.

[9] D. Youakim and P. Ridao, "Motion planning survey for autonomous mobile manipulators underwater manipulator case study," Robot. Auton. Syst., vol. 107, pp. 20–44, 2018.

[10] L. Janson, E. Schmerling, A. Clark, and M. Pavone, "Fast marching tree: A fast marching sampling-based method for optimal motion planning in many dimensions," Int. J. Robot. Res., vol. 34, no. 7, pp. 883–921, 2015.

[11] J. D. Gammell, S. S. Srinivasa, and T. D. Barfoot, "Batch informed trees (bit*): Sampling-based optimal planning via the heuristically guided search of implicit random geometric graphs," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2015, pp. 3067–3074.

[12] B. Ichter, J. Harrison, and M. Pavone, "Learning sampling distributions for robot motion planning," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2018, pp. 7087–7094.

[13] M. Zucker, N. Ratliff, A. D. Dragan, M. Pivtoraiko, M. Klingensmith, C. M. Dellin, J. A. Bagnell, and S. S. Srinivasa, "CHOMP: Covariant hamiltonian optimization for motion planning," Int. J. Robot. Res., vol. 32, no. 9-10, pp. 1164–1193, 2013.

[14] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "STOMP: Stochastic trajectory optimization for motion planning," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2011, pp. 4569–4574.

[15] W. Li and E. Todorov, "Iterative linear quadratic regulator design for nonlinear biological movement systems," in Proc. of the International Conference on Informatics in Control, Automation and Robotics (ICINCO), Setúbal, PT, 2004.

[16] R. Diankov, N. D. Ratliff, D. Ferguson, S. S. Srinivasa, and J. Kuffner, "Bispace planning: Concurrent multi-space exploration," in Proc. Robot. Sci. Syst. (RSS), Zurich, CH, 2008.

[17] D. Berenson, S. S. Srinivasa, D. Ferguson, A. Collet, and J. J. Kuffner, "Manipulation planning with workspace goal regions," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2009, pp. 618–624.

[18] I. A. Şucan and L. E. Kavraki, "Kinodynamic motion planning by interior-exterior cell exploration," in Algorithmic Foundation of Robotics VIII. Springer, 2009, pp. 449–464.

[19] A. Shkolnik and R. Tedrake, "Path planning in 1000+ dimensions using a task-space voronoi bias," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2009, pp. 2061–2067.

[20] M. Zucker, J. A. Bagnell, C. G. Atkeson, and J. Kuffner, "An optimization approach to rough terrain locomotion," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2010, pp. 3589–3595.

[21] S. Dalibard and J.-P. Laumond, "Control of probabilistic diffusion in motion planning," in Algorithmic Foundation of Robotics VIII. Springer, 2009, pp. 467–481.

[22] B. Ichter and M. Pavone, "Robot motion planning in learned latent spaces," IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 2407–2414, 2019.

[23] M. P. Do Carmo, Differential Geometry of Curves and Surfaces: Revised and Updated Second Edition. Courier Dover Publications, 2016.

[24] B. Zhang, J. Yang, J. Wu, D. Qin, and L. Gao, "PCA-subspace method—is it good enough for network-wide anomaly detection," in IEEE Network Operations and Management Symposium, 2012, pp. 359–367.

[25] G. Chen, H. Zhang, and C. Xiong, "Maximum margin dirichlet process mixtures for clustering," in AAAI Conference on Artificial Intelligence, 2016, pp. 1491–1497.

[26] D. Görür and C. E. Rasmussen, "Dirichlet process gaussian mixture models: Choice of the base distribution," Journal of Computer Science and Technology J. Comput. Sci. Tech-ch., vol. 25, no. 4, pp. 653–664, 2010.

[27] D.-H. Kim, Y.-S. Choi, T. Park, J. Y. Lee, and C.-S. Han, "Efficient path planning for high-DOF articulated robots with adaptive dimensionality," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2015, pp. 2355–2360.

[28] J. Nocedal and S. Wright, Numerical optimization. Springer Science & Business Media, 2006.

[29] J. Woolfrey, D. Liu, and M. Carmichael, "Kinematic control of an autonomous underwater vehicle-manipulator system (auvms) using autoregressive prediction of vehicle motion and model predictive control," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2016, pp. 4591–4596.

[30] M. Kang, D. Kim, and S.-E. Yoon, "Harmonious sampling for mobile manipulation planning," arXiv preprint arXiv:1809.07497, 2018.

[31] B. Jia, Z. Pan, and D. Manocha, "Fast motion planning for high-dof robot systems using hierarchical system identification," in Proc. of the IEEE Intl. Conf. on Robot. and Autom. (ICRA), 2019, pp. 5140–5147.

[32] W. Lu and D. Liu, "A scalable sampling-based optimal path planning approach via search space reduction," IEEE Access, vol. 7, pp. 153 921–153 935, 2019.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2020.2977379, IEEE Access

HUAN YU received his B.S. in Electronic and Information Engineering from Hunan University, Changsha, China in 2014. He then received his M.S. in Control Engineering from Beijing Institute of Technology, Beijing, China in 2016, where he is currently pursuing the Ph.D. degree. He is a two-years visiting research Ph.D. student in Centre for Autonomous Systems, University of Technology Sydney, Australia. His research interests include motion planning, localization and SLAM for autonomous robots.

WENJIE LU received his B.S. in Mechatronic Engineering from Zhejiang University, Hangzhou, China in 2009. He then received his M.S. and Ph.D. in Mechanical Engineering from Duke University, Durham, NC, USA in 2011 and 2014 respectively. His research interests include the learning of efficient motion planners and controllers for mobile robots that are subject to uncertainties in 3D complex environment. He has been developing such algorithms via integrating machine learning, approximate dynamic programming, hybrid systems, information theories, and nonparametric Bayesian models.

YONGQIANG HAN received his M.S. and Ph.D. degree in Control Science and Engineering from Beijing Institute of Technology, Beijing, China in 2008 and 2012 respectively. Now he is a lecturer in Automation School of Beijing Institute of Technology. His research interests include robot navigation, localization and control.

DIKAI LIU received his Ph.D. degree in 1997. His main research interest is robotics including exploration, motion planning, robot teams, and physical human-robot interaction. He has been developing novel methods and algorithms that enable robots to operate in unstructured and complex 3D environments autonomously or collaboratively with human users. Example robotic systems he developed and practically deployed include autonomous grit-blasting robots for steel bridge maintenance, assistive robots for human strength augmentation in industrial applications.

MIAO ZHANG received B.E. degree and M. S. degree from University of Science and Technology Beijing, China, in 2013 and 2016, respectively. He is currently a Ph.D. candidate at Beijing Institute of Technology, China, and University of Technology Sydney (UTS), Australia. His research interest are Auto Machine Learning, Deep Learning, and Bayesian Optimization.