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

Coupling Tensor Trains with Graph of Convex Sets: Effective Compression, Exploration, and Planning in the C-Space

Gerhard Reinerth[1], Riddhiman Laha[1,2], and Marcello Romano[1]

***Abstract*— We present TANGO (Tensor ANd Graph Optimization), a novel motion planning framework that integrates tensor-based compression with structured graph optimization to enable efficient and scalable trajectory generation. While optimization-based planners such as the Graph of Convex Sets (GCS) offer powerful tools for generating smooth, optimal trajectories, they typically rely on a predefined convex characterization of the high-dimensional configuration space—a requirement that is often intractable for general robotic tasks. TANGO builds further by using Tensor Train decomposition to approximate the feasible configuration space in a compressed form, enabling rapid discovery and estimation of task-relevant regions. These regions are then embedded into a GCS-like structure, allowing for geometry-aware motion planning that respects both system constraints and environmental complexity. By coupling tensor-based compression with structured graph reasoning, TANGO enables efficient, geometry-aware motion planning and lays the groundwork for more expressive and scalable representations of configuration space in future robotic systems. Rigorous simulation studies on planar and real robots reinforce our claims of effective compression and higher quality trajectories.**

I. Introduction

The problem of generalized motion planning for articulated systems, particularly under combined system and task-level costs and constraints, remains an open and active area of research in robotics [1]. In order to make it tractable, the following sub-areas have been studied: (a) Task assignment [2]–[4], (b) Path and trajectory planning [5], [6], and (c) Collision avoidance [7]–[11]. Notwithstanding, region generation in the C-space remains a roadblock [12]. Researchers have proposed various perspectives to tackle this challenge. Decomposition methods for handling non-convex shapes vary significantly in strategy. Iterative approaches, such as that of Lien and Amato [13], repeatedly partition the shape by removing the largest concavity, gradually improving convexity. Optimization-based methods, like Liu et al. [14], instead formulate the problem as a mixed-integer program, identifying cutting planes that minimize concavity under a prescribed threshold. Clustering-based techniques, exemplified by [15], group faces of the shape into clusters that together approximate convex components. While differing in methodology—iterative refinement, optimization-driven cuts, or face clustering—all of these techniques return only approximately convex partitions that cover the original geometry. A key drawback is that using the convex hulls of these components in motion planning may inflate feasible regions and inadvertently intersect with obstacles, undermining safety guarantees.

As robots move to more complex tasks, sampling becomes non-trivial and, at times, intractable [16], [17]. Extremely simple but ubiquitous tasks like rotating a crank or moving a cup of water still involve sampling from a constrained differentiable manifold [18]. More specifically, continuous constraint functions need to be constructed within the configuration space $\mathcal{Q}$. Sampling-based planners operating in the ambient C-space, while task or system constraints are often defined as a set of equality conditions that specify a zero-measure subset [19]. As a result, uninformed random samples have zero probability of satisfying these constraints [20]. To the best of the authors' knowledge, no constrained sampling-based planner has utilized a planning methodology that employs a coverage estimate in its planning process.

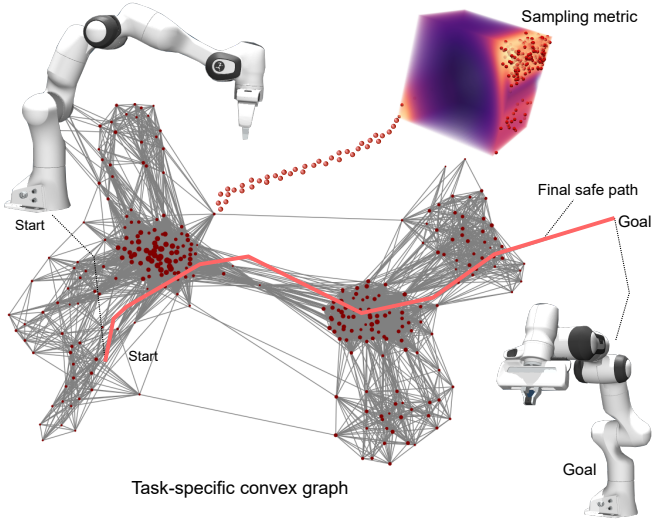

Fig. 1. Our algorithm enables a task-specific sampling metric for approximating the feasible configuration space for a system. These approximated regions are then used for discovering a convex structured graph for effective motion planning within the actuator bounds. Note that the larger the size of the node, the larger is the volume of the corresponding convex set.

In this work, therefore, we take a different approach. Rather than sampling naively in all dimensions, we attempt to subsample the configuration space where feasible motions are likely to exist using a greedy compression technique [21]. Our central hypothesis is that effective compression enables more targeted and efficient exploration of the space. This technique also facilitates the sampling of both good (feasible) and bad (infeasible) configurations, allowing us to characterize the configuration space more thoroughly as shown in Fig. 1. Once characterized, cost functions for particular planning tasks can be reformulated as a probability density function. Ultimately, convex regions are grown in the configuration space. In other words, we try to identify large, safe convex regions within the free space, which can be used to construct the GCS and plan through it efficiently. Safety, in this context, refers to maintaining configurations away from singularities while

[1]Gerhard Reinerth, Riddhiman Laha, and Marcello Romano are with the Technical University of Munich, Germany g.reinerth@tum.de, marcello.romano@tum.de
[2]Riddhiman Laha is also with Northeastern University, Boston, USA r.laha@northeastern.edu

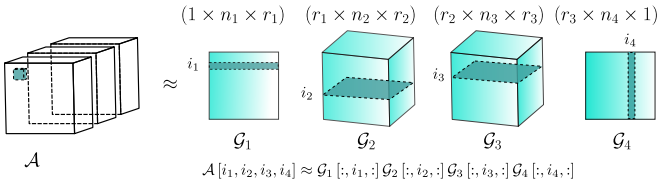

Fig. 2. Representation of high-dimensional space using TT approximation. Note the decomposition into the different tensor cores.

ensuring that all joint motions remain within their admissible limits.

Our main contributions are the following:

- We establish a principled way to explore the configuration space of complex systems using the tensor train decomposition.
- We enhance planning performance within a popular convex relaxation algorithm in the context of finding the shortest paths.
- We verify experimentally that the identified shortest path stays within the desired configuration space and has higher manipulability than traditional sampling-based planners.

II. Mathematical Preliminaries

The core idea here is to approximate a probability density function using the Tensor Train Decomposition (TTD), the goal being instantaneous retrieval of solutions. Next, we describe the concept of discovering convex regions in the C-space.

A. Tensor Train Decomposition

The TTD introduced by [22] decomposes a higher-order Tensor into low-rank tensor cores. We briefly introduce the basic notation and operations by slightly using MATLAB notation for convenience.

A tensor $\mathcal{A} \in \mathbb{R}^{n_1 \times n_2 \times \cdots \times n_d}$ is decomposed in tensor cores $\mathcal{G}_k \in \mathbb{R}^{r_{k-1} \times n_k \times r_k}$:

$$\mathcal{A}[i_1, i_2, \cdots, i_d] \approx \mathcal{G}_1[:, i_1, :]\,\mathcal{G}_2[:, i_2, :] \cdots G_d[:, i_d, :] \quad (1)$$

where $i_1, i_2, \cdots i_d$ denote index entries of tensor $\mathcal{A}$. $\mathcal{G}_k[:, i_k, :]$ denotes the i_k-th slice of tensor core $\mathcal{G}_k$. Note that $r_0 = r_d = 1$. Assuming that each tensor core has rank $r_k = r$ and dimension $n_k = n$, the required number of parameters may be estimated as $P = 2nr + (d-2)\,nr^2$. Thus, the number of parameters scale linearly with the dimensionality and quadratically with the rank of the cores.

The TTD also provides basic operations in the TT format, such as elementwise addition. Suppose two tensors $\mathcal{A}, \mathcal{B} \in \mathbb{R}^{n_1 \times n_2 \cdots \times n_d}$ have decompositions $\mathcal{A} \approx \mathcal{G}_1^{\mathcal{A}} \cdots \mathcal{G}_d^{\mathcal{A}}$ and $\mathcal{B} \approx \mathcal{G}_1^{\mathcal{B}} \cdots \mathcal{G}_d^{\mathcal{B}}$. Then the tensor cores $\mathcal{C} = \mathcal{A} \oplus \mathcal{B}$ can be expressed with tensor cores of $\mathcal{A}$ and $\mathcal{B}$ s.t.

$$\begin{aligned}
\mathcal{G}_1^{\mathcal{C}}[:, i_1, :] &= \begin{pmatrix} \mathcal{G}_1^{\mathcal{A}}[:, i_1, :] & \mathcal{G}_1^{\mathcal{B}}[:, i_1, :] \end{pmatrix}, \\
\mathcal{G}_d^{\mathcal{C}}[:, i_d, :] &= \begin{pmatrix} \mathcal{G}_d^{\mathcal{A}}[:, i_d, :] \\ \mathcal{G}_d^{\mathcal{B}}[:, i_d, :] \end{pmatrix}, \\
\mathcal{G}_k^{\mathcal{C}}[:, i_k, :] &= \begin{pmatrix} \mathcal{G}_k^{\mathcal{A}}[:, i_k, :] & 0 \\ 0 & \mathcal{G}_k^{\mathcal{B}}[:, i_k, :] \end{pmatrix}
\end{aligned} \quad (2)$$

This operation leads to an increase in rank, which may be alleviated by also utilizing the *rounding operation* described in [22].

B. PDF approximation by TT-Cross

The TT-Cross algorithm [23], [24] is used to approximate high-dimensional tensors, which otherwise would become intractable to store in computer memory due to the curse of dimensionality. TT-Cross employs a greedy sampling strategy over the high-dimensional object and iteratively refines the tensor cores, with the corresponding rank updates being determined adaptively by the algorithm itself. It is therefore suitable for approximating (high-dimensional) black-box functions. In our case, TT-Cross is utilized to explore the configuration space of a robotic manipulator. For identifying certain configurations of a robot manipulator, we replace a task-specific cost function or metric with a probability density function.

C. Discovering Convex Sets for Planning

We now want to describe the free C-space such that a convex characterization is possible. Iterative Regional Inflation by Semi-definite programming (IRIS), first introduced in [6], begins by selecting a seed point in C_{free} and iteratively inflates a convex polytope around it that remains entirely within the free space. Directly maximizing the volume of a convex polytope (represented as an intersection of halfspaces) is computationally intractable, as the volume of such polytopes is P-hard to compute. Instead, IRIS maximizes the volume of the largest ellipsoid that can be inscribed within the polytope as a proxy for region size.

The optimization problem is bi-convex in the decision variables defining the separating hyperplanes (halfspaces) and those defining the ellipsoid. Although not jointly convex, the problem can be efficiently solved via alternating optimization: one alternates between (i) pushing the separating hyperplanes outward to exclude nearby obstacles (the SeparatingHyperplanes step), and (ii) expanding the ellipsoid to fit maximally within the current polytope (the InscribedEllipsoid step) [6]. Through this iterative process, IRIS converges to a locally maximal convex region that is certified to be free of collisions.

Finding the shortest path in C-space is often approached by constructing and searching a graph, typically through sampling. To this end, local segments can be joined by shortest path queries once a dense global graph has been built. As mentioned in the introduction, another approach to constructing paths is using local optimization. This naturally raises the question of how best to balance dense sampling with effective optimization in order to approach true optimality. One way to look at this problem is through the lens of convexity. Convexity provides a unifying principle for relating both sets and functions. Once the problem is formalized as a convex program, specialized optimization tools can be employed for global optimality. One such framework that has been introduced in the recent past is the Graph of Convex Sets [25], [26] (GCS). We make use of a variant of GCS - the shortest path problem (SPP) - to find a geometric path inside convex regions connecting the start and target pose.

III. Problem Formulation

The planning problem that we focus on is the following: *Given a start and a target configuration, and a sampling metric, we seek to identify and grow safe convex regions within compressed, low-rank tensor representation of the*

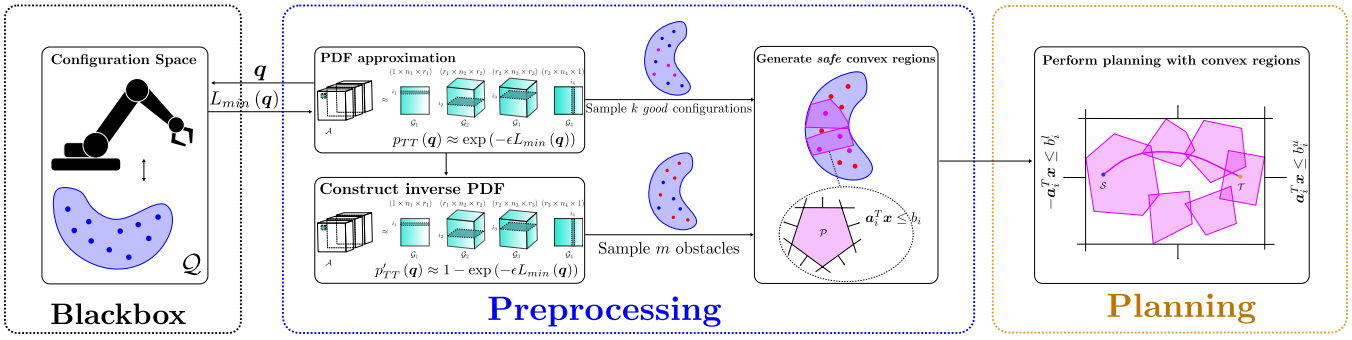

Fig. 3. An overview of our TANGO algorithm is illustrated as follows. We begin by sampling the configuration space and constructing an inverse probability density function (PDF) using a chosen task metric L_{min}. From this distribution, samples are drawn and classified into feasible and infeasible categories. These classifications, along with IRIS, enable the discovery of safe convex regions within the configuration space. Finally, a shortest-path search over the discovered convex sets yields the resulting trajectory from start to goal configuration.

configuration space. These regions serve as the foundation for planning a smooth, shortest path that adheres to the system's mechanical constraints.

More formally, let $\mathcal{M} : \mathbb{R}^m \to \mathbb{R}_{\geq 0}$ denote a task-specific *sampling metric*, which assigns a non-negative cost $C(q)$ to each configuration $\boldsymbol{q} \in \mathbb{R}^m$. Given a start configuration $\boldsymbol{q}_s \in \mathbb{R}^m$, a target configuration $\boldsymbol{q}_t \in \mathbb{R}^m$, and the sampling metric $\mathcal{M}$, our goal is to identify a compressed representation of the feasible configuration space that preserves safety and tractability. To this end, we approximate the density of C_{fes}

$$p(q;\gamma) = \exp(-\gamma C(q)), \tag{3}$$

using *TT-Cross* to efficiently encode high-dimensional feasibility information. From this representation, we seek to *discover convex regions* in the configuration space and embed them into a *Graph of Convex Sets (GCS)*.

The planning objective is then to compute a smooth, collision-free path π

$$\pi : [0,1] \to \mathbb{R}^m, \quad \pi(0) = \boldsymbol{q}_s, \;\; \pi(1) = \boldsymbol{q}_t, \tag{4}$$

that respects both the *mechanical bounds* of the system and the *geometric constraints* of the identified convex regions, while minimizing a cost functional (e.g., path length, energy, or task-specific metrics).

IV. Task-Specific Sampling Metrics

The metric $\mathcal{M} : \mathbb{R}^m \to \mathbb{R}_{\geq 0}$ has to be chosen depending on the task at hand. In general, we want to minimize some cost function $\psi(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}$ that encodes a suitable performance criterion. For exposition purposes, we briefly describe two metrics that we study in the work: (i) the Yoshikawa Manipulability Metric, and (ii) a Riemannian Metric.

The Yoshikawa manipulability metric provides a quantitative measure of how dexterous a manipulator is at a given configuration [27]. It is defined as the volume of the velocity ellipsoid induced by the Jacobian, i.e.,

$$C(\boldsymbol{q}) = \sqrt{\det(\boldsymbol{J}(\boldsymbol{q})\boldsymbol{J}(\boldsymbol{q})^\top)}, \tag{5}$$

where $\boldsymbol{J}(\boldsymbol{q})$ is the manipulator Jacobian. High values of $C(\boldsymbol{q})$ correspond to configurations with greater ability to generate end-effector motions in arbitrary directions, while low values indicate kinematic singularities or restricted mobility. In our framework, we treat $C(\boldsymbol{q})$ as a task-specific sampling metric to bias the TT-Cross decomposition toward more dexterous regions of the configuration space. This allows us to capture manipulability-aware feasibility information in the compressed representation.

For the next metric, we adopt a geometry-aware singularity index ξ, defined as the squared Riemannian distance between the manipulability ellipsoid $\boldsymbol{M}(\boldsymbol{q}) = \boldsymbol{J}\boldsymbol{J}^T$ and a reference ellipsoid Σ, i.e.,

$$\xi(\boldsymbol{q}) = \left\| \log\left(\Sigma^{-1/2}\boldsymbol{M}\Sigma^{-1/2}\right) \right\|_F^2 \tag{6}$$

We assume that the ellipsoid Σ can be regarded as a hypersphere, representing the optimal configuration of a manipulator. Unlike the traditional Yoshikawa metric, which only considers the volume of the manipulability ellipsoid, $\xi(\boldsymbol{q})$ encapsulates its full geometry—including size, shape, and orientation. This comprehensive characterization enables more accurate detection of singularities and better-informed sampling of the configuration space. Since the Yoshikawa index can remain unchanged even when the ellipsoid undergoes significant deformation, it may fail to reflect critical changes in manipulability. In contrast, $\xi(\boldsymbol{q})$ is sensitive to such changes due to its use of a Riemannian metric that respects the affine-invariant structure of symmetric positive definite matrices [28]. This makes $\xi(\boldsymbol{q})$ a more reliable and robust tool for guiding motion planning and singularity avoidance in complex robotic systems.

V. Tensor and Graph Optimization (TANGO)

Our approach can be split into two stages as elucidated in Fig. 3, which are required to perform path planning within safe regions of the robot configuration space.

A. Tensor Train Preprocessing

The preprocessing stage requires a PDF $p(\boldsymbol{q};\gamma)$ from the *blackbox* system, which in our case is a robotic manipulator. As in Shetty et al. [29], a metric or cost function is employed to construct a PDF. Good configurations of the robot manipulator are assumed to achieve a high likelihood; singular configurations should have a small likelihood. Using TT-Cross, we approximate a high-dimensional PDF p_{TT}, which represents preferable configurations. By using Equation 2, the inverse $p'_{TT} = 1 - p_{TT}$ can be constructed in the TT format. After both PDFs are available in the TT format, we obtain *initial-* and *obstacle* configurations, which can be efficiently sampled in the TT format [30].

Algorithm 1 Tensor Train Preprocessing

procedure PREPROCESS(p)
  $\mathcal{A} \leftarrow$ Approximate $p(\boldsymbol{q}, \gamma)$ using TT-Cross once;
  $\tilde{\mathcal{A}} \leftarrow$ Construct inverse TT-PDF using Equation 2;
  $\boldsymbol{C}_k \leftarrow$ Draw k_c samples from $\mathcal{A}$ using TT-Sample;
  $\boldsymbol{C}_n \leftarrow$ Draw n_o samples from $\tilde{\mathcal{A}}$ using TT-Sample;
  $\boldsymbol{C}_{k'} \leftarrow$ Select best k_c' configurations from $\boldsymbol{C}_k$;
  $\boldsymbol{C}_{n'} \leftarrow$ Select best n_o' obstacles from $\boldsymbol{C}_n$;
  OBST $\leftarrow$ Construct convex obstacles from $\boldsymbol{C}_{n'}$;
  SAFE $\leftarrow$ Perform IRIS using OBST and $\boldsymbol{C}_{k'}$;
return SAFE;

Algorithm 2 Planning using TANGO

procedure TANGO($p, \boldsymbol{q}_{start}, \boldsymbol{q}_{goal}$)
  $C_{safe} \leftarrow$ Execute Preprocess(p) and prune ▷ static, initialized only once;
  $\boldsymbol{S} \leftarrow$ Compute all possible intersections from C_{safe}; ▷ static, initialized only once;
  WP $\leftarrow \emptyset$
  if $\boldsymbol{q}_{start}, \boldsymbol{q}_{goal}$ in C_{safe} **then**
    GCS $\leftarrow$ Construct GCS using $\boldsymbol{S}$;
return WP;

For computing safe convex sets with IRIS [7], the obstacle configurations need to be converted to convex sets. We employ RNNDBSCAN clustering to merge obstacles into convex sets [31]. Algorithm 1 summarizes the overall procedure.

B. Shortest Path Planning

The safe convex set candidates from the preprocessing stage are further pruned, since safe convex sets may reside within another convex set. These special cases won't contribute to the planning stage and introduce additional complexity during the later optimization procedure within GCS. The remaining sets are now used to construct the general graph structure for GCS. For each pair of the remaining convex sets, we check for intersections. Each intersection then represents an edge within the GCS structure.

Finally, the resulting GCS may now be used for planning (c.f. Algorithm 2).

C. Shortest Path Problem in Graph of Convex Sets (GCS)

We consider the shortest path problem (SPP) formulated over a *Graph of Convex Sets* (GCS). Let $G := (\mathcal{V}, \mathcal{E})$ be a directed graph, where:

- Each vertex $v \in \mathcal{V}$ is associated with a non-empty compact convex set $\mathcal{X}_v \subset \mathbb{R}^n$,
- Each vertex also contains a continuous decision variable $\mathbf{x}_v \in \mathcal{X}_v$,
- Each edge $e = (u, v) \in \mathcal{E}$ is associated with a convex cost function $\ell_e(\mathbf{x}_u, \mathbf{x}_v) : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}_{\geq 0} \cup \{\infty\}$.

A path $p = (v_0, \ldots, v_K)$ is a sequence of distinct vertices such that:

$$v_0 = s, \quad v_K = t, \quad (v_k, v_{k+1}) \in \mathcal{E}, \quad \forall k = 0, \ldots, K-1.$$

We denote the set of edges traversed by this path as:

$$\mathcal{E}_p := \{(v_0, v_1), \ldots, (v_{K-1}, v_K)\},$$

and let $\mathcal{P}$ denote the set of all valid s-t paths in the graph G.

The shortest path problem over GCS is then defined as:

$$\begin{aligned} \text{minimize} \quad & \sum_{e=(u,v)\in\mathcal{E}_p} \ell_e(\mathbf{x}_u, \mathbf{x}_v) \\ \text{subject to} \quad & p \in \mathcal{P}, \\ & \mathbf{x}_v \in \mathcal{X}_v, \quad \forall v \in p. \end{aligned} \tag{7}$$

Here, the decision variables are both:

- the discrete path $p \in \mathcal{P}$, and
- the continuous vertex configurations $\mathbf{x}_v \in \mathcal{X}_v$.

Constraining the path to traverse vertices $\mathcal{X}_v$ inherently enables avoiding self-collisions, which may be encountered in path planning with robotic manipulators.

Edge Cost Functions: Two common choices for edge cost functions are:

a) Euclidean Distance (Path Length):

$$\ell_e(\mathbf{x}_u, \mathbf{x}_v) := \|\mathbf{x}_v - \mathbf{x}_u\|_2. \tag{2.2}$$

b) Squared Euclidean Distance (Energy-like):

$$\ell_e(\mathbf{x}_u, \mathbf{x}_v) := \|\mathbf{x}_v - \mathbf{x}_u\|_2^2. \tag{2.3}$$

Feasibility via Convex Edge Constraints: To enforce edge feasibility, one can define a convex constraint set $\mathcal{X}_e \subset \mathbb{R}^{2n}$ and set:

$$\ell_e(\mathbf{x}_u, \mathbf{x}_v) = \infty \quad \text{if } (\mathbf{x}_u, \mathbf{x}_v) \notin \mathcal{X}_e.$$

This formulation allows us to encode motion constraints or dynamic feasibility into the edge cost.

VI. Implementation Details

To accelerate computations, we utilize multiprocessing, where possible. We carefully select operations that are eligible for parallel computations.

IRIS: Computing a large number of convex sets using IRIS easily becomes a computational bottleneck, especially when several convex candidates need to be computed. We assume that the initial configurations sampled by TT can be in a close neighborhood. After computing a batch of initial safe convex sets, we check if some of the remaining configurations are covered by the convex sets/polyhedra. Candidates that are already covered are then not considered for further computations. Therefore, the resulting computational complexity may be, in the best case, linear, and in the worst case, still quadratic.

Pruning: Pruning the candidate convex sets can also, in general, result in quadratic runtimes, since checking for each possible intersection requires in total $n(n-1)/2$ combinations. To accelerate the pruning procedure, we first sort the convex set candidates according to their volume in descending order. Sets, which are contained in the sets of a bigger volume, or which are mostly covered by a set of greater volume, are removed from the potential candidate sets.

VII. Experimental Validation

We demonstrate the effectiveness of our TANGO algorithm using 3 distinct perspectives. Our main objective in these simulation studies is to show that our proposed framework is scalable and can be used to generate high quality trajectories for physical systems. In addition, we also elucidate the TANGO memory footprint.

TABLE I

TUNABLE PARAMETERS IN TANGO.

Notation	Corresponding Description
k_c	Number of initial configurations
k'_c	Number of k'_c best configurations
n_o	Number of obstacle configurations
n'_o	Number of n'_o best obstacle configurations
γ	Scaling factor for cost function/metric
n_{nswp}	Number of TT-Cross iterations
n_{IRIS}	Number of IRIS iterations

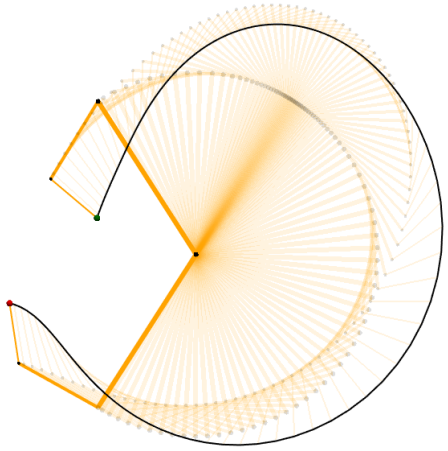

Fig. 4. Executed robot trajectory from start configuration (red sphere) to goal configuration (green sphere) using TANGO. It should be noted that, due to the presence of joint limits, the geodesically shortest path in the configuration space does not necessarily correspond to the shortest path in the task (operational) space.

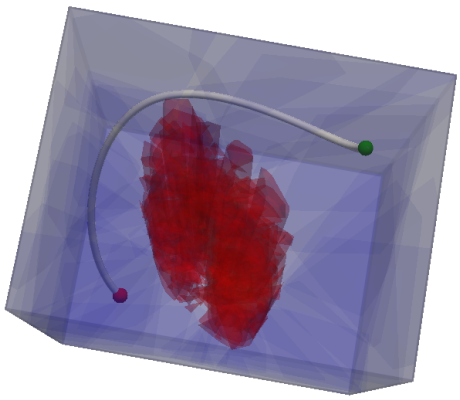

Fig. 5. Safe convex sets and path planning. The blue convex sets are used for planning with TANGO, and the red convex sets represent obstacles in the configuration space. The final path is shown in white.

During the illustrative example and the experiment, we utilize tunable parameters as represented in Table VII.

The parameters used are described in the corresponding (sub)-sections.

A. Illustrative Analysis: 3-DoF Planar Manipulator

To evaluate the effectiveness of the proposed framework, we begin with an illustrative case study involving a simple 3 DoF planar robotic manipulator (taken from Lück et al. [32]), designed to move its end-effector in a 2D workspace.

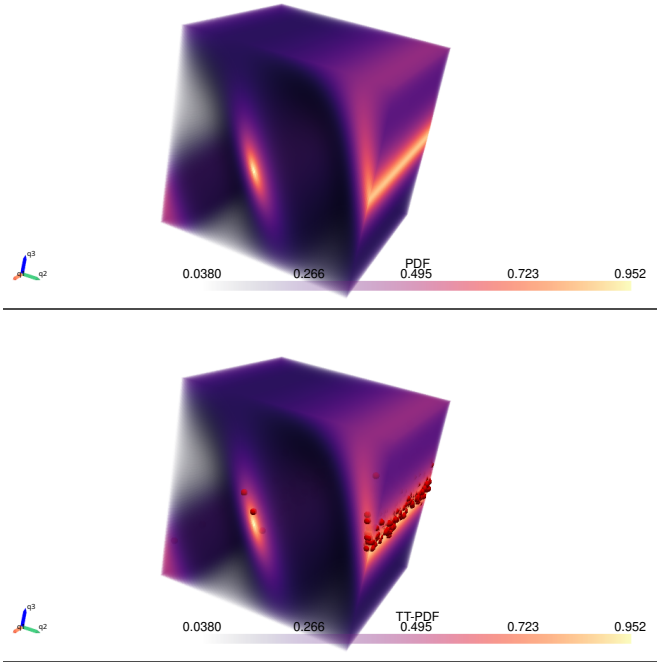

Fig. 6. Original pdf (top) and TT-Cross approximation (bottom) for the Yoshikawa metric. The red points (bottom picture) denote samples taken from the TTD. Regions with a higher probability are sampled more likely. For illustrative and visualization purposes, we fixed the scaling parameter value to $\gamma = 0.1$.

First, we need to decide on which metric is to be utilized for constructing a PDF used for sampling. We consider the classic Yoshikawa metric and the Riemannian metric (c.f Equation 5, 6). We show the PDFs and their corresponding approximations in Figure 6 and Figure 8. The planar 3DoF manipulator is a non-escapable singular configuration (c.f. [32]) when fully stretched. The singular configurations appear as low PDF values occuring in roughly in the center of the Figures. After sampling from both PDFs, we clearly see that it is likely to happen to sample near-singular configurations when constructing a PDF using Equation 5. Therefore, from here on, we will just consider the Riemmanian metric.

Continuing our illustrative example we demonstrate TANGO's core components—tensor-based feasibility modeling, convex region generation, and graph-based trajectory planning—under controlled conditions.

We discretize each joint of the 3 DoF manipulator into 128 bins, resulting in a dense 3D tensor $\mathcal{A} \in \mathbb{R}^{128\times128\times128}$, which encodes a task-specific feasibility metric. To enhance computational tractability, we reshape this tensor into a 7D structure $\mathcal{A}' \in \mathbb{R}^{8\times8\times\cdots\times8}$, enabling efficient compression via TT-Cross decomposition.

Figures 7 and 8 show both the full feasibility tensor and its TT-approximated counterpart. To generate the inverse feasibility field, we use the element-wise TT subtraction operation:

$$\mathcal{A}_{\text{inv}} = \mathbb{1} \ominus \mathcal{A}',$$

where $\mathbb{1}$ is a unit-valued tensor and $\ominus$ denotes element-wise subtraction in TT format.

For environment modeling, we sample 1.5×10^4 configurations, including both feasible and obstacle-occupied points. From these, we select the top 10^4 obstacle configurations and

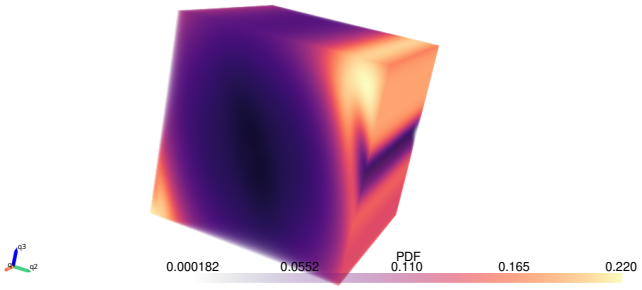

Fig. 7. Original PDF constructed using the Riemannian metric ($\gamma = 1.0$). The Riemannian metric results in comparatively more favorable manipulability configurations, whereas the Yoshikawa metric induces a more restricted spatial domain characterized by high-manipulability regions.

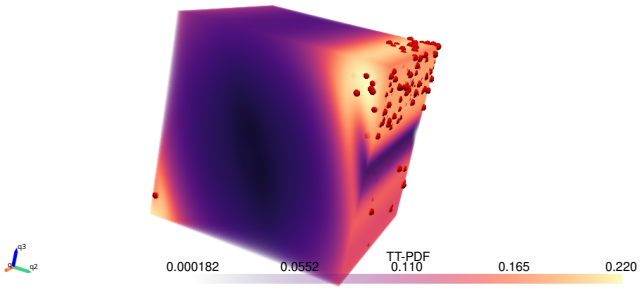

Fig. 8. TT-Cross approximation of the PDF. The red points here denote samples taken from the TTD. Regions with a higher probability are sampled more likely.

500 high-quality feasible samples based on the task-specific metric. The obstacle configurations are clustered using RNN-DBSCAN, producing convex obstacle regions, while safe regions are extracted using the IRIS algorithm and pruned as described in Section VI.

The final Graph of Convex Sets (GCS) is constructed by identifying intersecting convex regions, enabling shortest-path planning through safe corridors. Figure 5 illustrates a planned trajectory across convex sets. This example assumes no self-collision, focusing instead on validating TANGO's ability to identify and traverse meaningful configuration space regions under task-specific metrics.

B. Scalability and Trajectory Generation for Panda Arm

To demonstrate the scalability and trajectory quality of our proposed method, we now evaluate its performance on a high-dimensional manipulator model—a 7-DoF Franka Emika Panda robot [33]. To reduce experimental complexity while still retaining meaningful motion planning challenges, we lock joints $5, 6$, and 7, resulting in a 4-DoF configuration space. Extending the framework to seven degrees of freedom (7DoF) is expected to increase computational complexity, as the safe convex sets—represented as polyhedra—will, in general, exhibit greater geometric complexity. Consequently, in the present work, we focus on a lower-dimensional, simplified problem setting, with the intention of subsequently generalizing the approach to systems with a larger number of degrees of freedom in future research.

We discretize the 4D space into 128 equally spaced bins per joint, forming a tensor $\mathcal{A} \in \mathbb{R}^{128\times128\times128\times128}$. For efficient high-dimensional compression, the tensor is reshaped into a 14D tensor $\mathcal{A}' \in \mathbb{R}^{4\times4\times\cdots\times4}$, which is then approximated using the TT-Cross decomposition technique. It should be acknowledged that alternative choices for the tensor dimensionality are feasible, and that the determination of both an optimal number of dimensions [34] and an appropriate ordering of these dimensions [35] lies beyond the scope of the present study. The specific dimensional configuration employed in this work was determined empirically. TT-Cross approximates the PDF within $n_{nswp} = 30$ iterations. We construct the PDF by scaling the Riemannian Metric (6) with a factor $\gamma = 0.1$.

Following the feasibility approximation, we construct the inverse feasibility tensor and sample $n_o = k_c = 5 \times 10^4$ configuration points. From these, we select the best $k_c' = 10^3$, $n_o' = 10^4$ samples from each set based on the Riemannian metric. Configurations with self-collisions among the initial candidates are reclassified as obstacles. Convex obstacle regions and safe convex sets are then extracted using the same pipeline as in Section VII-A. Figure 1 shows the convex regions connected using a graph. We perform $n_{IRIS} = 2$ IRIS iterations for the convex set computations. This ensures that the convex sets stay in the neighborhood of their initial configurations.

To evaluate trajectory quality, we compare TANGO to the standard RRT planner. We randomly select 100 start–goal pairs with large joint-space distances to ensure non-trivial planning tasks. Both planners generate collision-free paths, which are subsequently refined using TOPPRA [36] to yield smooth, dynamically feasible trajectories. We evaluate each trajectory and measure the worst-case possible PDF-Score. This gives us an empirical lower bound estimate of the performance of each of the approaches.

As shown in Figure 10, TANGO consistently produces more structured and concise plans—typically requiring only 3–4 waypoints before refinement—resulting in significantly smoother trajectories than those produced by RRT. In addition, our approach yields a higher minimum PDF-score during each trajectory on average. This experiment highlights TANGO's ability to scale to higher-dimensional systems while maintaining both computational efficiency and trajectory quality.

C. Time & Memory Footprints

To further evaluate the practicality of TANGO, we analyze its computational efficiency with respect to two critical bottlenecks in robotics software: execution time and memory consumption. These metrics are particularly important in resource-constrained systems, where planners must operate under strict latency requirements while maintaining a small memory footprint. By systematically profiling both dimensions, we assess how well our method scales in comparison to existing baselines and whether it remains viable for real-world deployment on embedded or low-power platforms.

Table II compares the memory footprint and execution time of the TT-Cross algorithm when applied to tensors with original versus reshaped dimensions, averaged over 15 trials. The reshaped tensor (4^{14} dimensions) demonstrates a significantly reduced memory footprint compared to the original tensor (128^4 dimensions). Specifically, the mean number of parameters drops from approximately 4.36×10^4 to 7.11×10^3, with consistently lower variability, as indicated by a smaller standard deviation. Execution time also improves

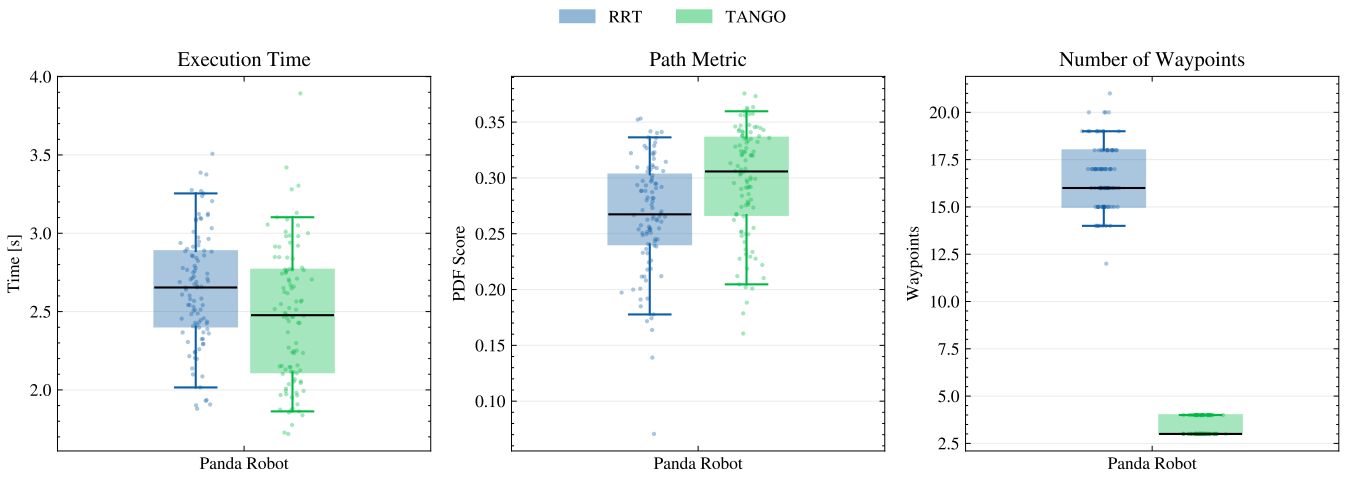

Fig. 9. Performance evaluation of TANGO and RRT. Execution times of executed trajectories (left), PDF-Score (middle), and number of resulting waypoints (right). The execution times of the generated trajectories are, on average higher for TANGO, while having a better PDF Score (higher values indicate better performance). TANGO requires in general, fewer waypoints, leading to smooth trajectories after further refinement.

substantially: the mean time decreases from 27.79 seconds for the original tensor to just 5.53 seconds for the reshaped version. Minimum and maximum values in both metrics further confirm this trend, highlighting that the reshaped tensor yields faster and more memory-efficient performance across all trials.

TABLE II

COMPARISON OF TT-CROSS MEMORY FOOTPRINT AND EXECUTION TIME FOR ORIGINAL AND RESHAPED TENSOR DIMENSIONS OVER 15 TRIALS.

Metric	**Original Dim.** 128^4	**Reshaped Dim.** 4^{14}
Memory Footprint (Parameters)		
Mean (μ)	4.36×10^4	7.11×10^3
Std Dev (σ)	2.25×10^4	1.70×10^3
Min	1.08×10^4	5.28×10^3
Max	6.66×10^4	1.19×10^4
Execution Time (s)		
Mean (μ)	27.7924	5.5336
Std Dev (σ)	2.6981	0.5166
Min	20.7543	4.6642
Max	33.6748	6.4782

VIII. CONCLUSION AND FUTURE WORK

This work presents a principled and efficient approach to motion planning by leveraging a greedy compression technique to focus sampling within promising regions of the configuration space. By targeting areas where feasible motions are more likely, our method avoids the inefficiencies of uniform sampling across all dimensions and enables a more thorough characterization of both feasible and infeasible configurations. This, in turn, allows us to reformulate task-specific cost functions as probability density functions and grow large, safe convex regions within the free space.

These convex regions form the foundation for constructing a Graph of Convex Sets (GCS), enabling fast and reliable planning. Our contributions demonstrate the effectiveness of this approach in improving planning performance, particularly in convex relaxation-based shortest path problems. Through experimental validation, we also confirmed that the computed paths remain within the intended feasible regions, further supporting the practical value of our method.

Although task-space obstacles are not explicitly addressed in the present work, a direct extension is readily attainable. Specifically, one may construct an additional tensor train either from Signed Distance Functions (SDFs) [21], [37] or from distance functions defined in the manipulator's configuration space and modeled as a probability density function (PDF). Then the two tensor trains can be combined in the preprocessing step. Further research is necessary to extend the proposed methodology to systems of higher dimensionality. In particular, the choice of the metric employed for constructing the probability density function (PDF), as well as the overall dimensionality of the tensor representation, is expected to substantially influence both the memory requirements and the attainable approximation accuracy. Further, topological connections could be analyzed between special manifolds in the configuration space and the tensor compressions. Lastly, the trajectory smoothness can be improved by considering spline segments to join the convex sets during planning.

REFERENCES

[1] Z. Kingston, M. Moll, and L. E. Kavraki, "Sampling-based methods for motion planning with constraints," *Annual review of control, robotics, and autonomous systems*, vol. 1, no. 1, pp. 159–185, 2018.

[2] A. Richards and J. How, "Mixed-integer programming for control," in *Proceedings of the 2005, American Control Conference, 2005.* IEEE, 2005, pp. 2676–2683.

[3] M. Alighanbari, Y. Kuwata, and J. P. How, "Coordination and control of multiple uavs with timing constraints and loitering," in *Proceedings of the 2003 American Control Conference, 2003.*, vol. 6. IEEE, 2003.

[4] Y. Zhang, R. Su, C. Sun, and Y. Zhang, "Modelling and traffic signal control of a heterogeneous traffic network with signalized and non-signalized intersections," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2017, pp. 1581–1586.

[5] B. Cetin, M. Bikdash, and F. Hadaegh, "Hybrid mixed-logical linear programming algorithm for collision-free optimal path planning," *IET Control Theory & Applications*, vol. 1, no. 2, pp. 522–531, 2007.

[6] R. Deits and R. Tedrake, "Footstep planning on uneven terrain with mixed-integer convex optimization," in *2014 IEEE-RAS international conference on humanoid robots*. IEEE, 2014, pp. 279–286.

[7] ——, "Computing large convex regions of obstacle-free space through semidefinite programming," in *Algorithmic Foundations of Robotics XI: Selected Contributions of the Eleventh International Workshop on the Algorithmic Foundations of Robotics*. Springer, 2015, pp. 109–124.

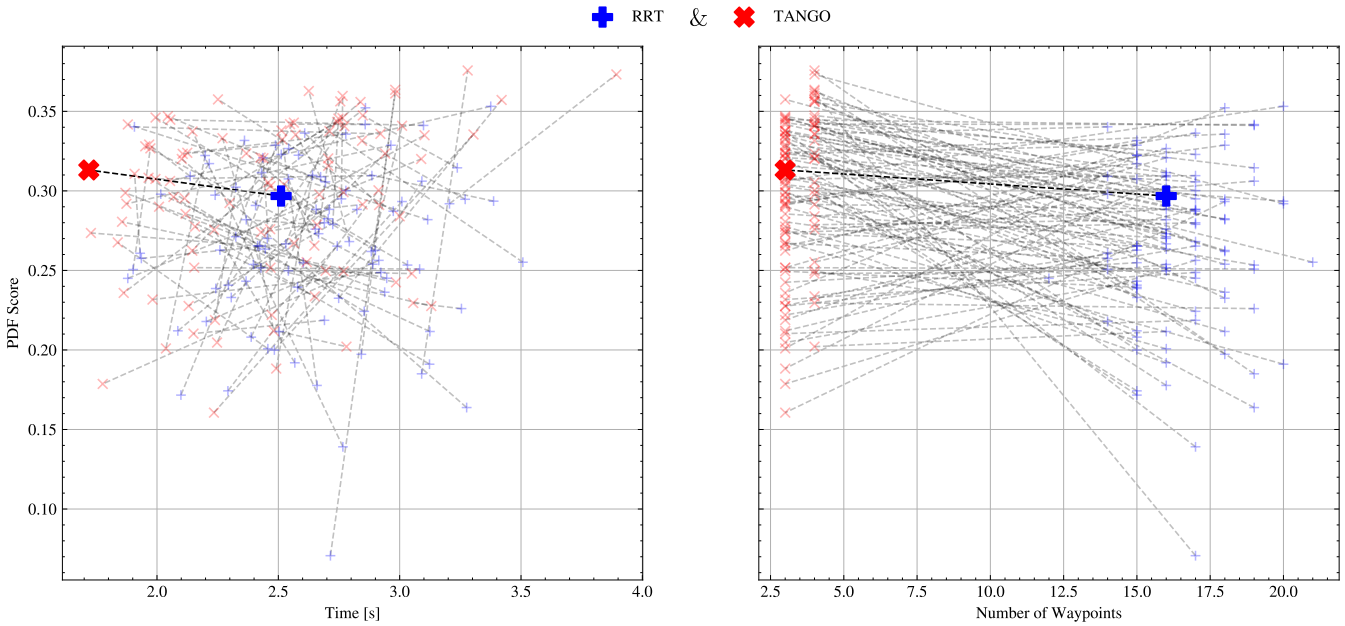

Fig. 10. Comparison between a naive RRT and TANGO. The plots show that our Riemannian sampling strategy in TANGO leads to approximate convergence in less time (left) and fewer waypoints (right) for different runs.

[8] R. Laha, L. F. Figueredo, J. Vrabel, A. Swikir, and S. Haddadin, "Reactive cooperative manipulation based on set primitives and circular fields," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 6577–6584.

[9] R. Laha, M. Becker, J. Vorndamme, J. Vrabel, L. F. Figueredo, M. A. Müller, and S. Haddadin, "Predictive multi-agent-based planning and landing controller for reactive dual-arm manipulation," *IEEE Transactions on Robotics*, vol. 40, pp. 864–885, 2023.

[10] P. Werner, A. Amice, T. Marcucci, D. Rus, and R. Tedrake, "Approximating robot configuration spaces with few convex sets using clique covers of visibility graphs," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 10 359–10 365.

[11] P. Werner, T. Cohn, R. H. Jiang, T. Seyde, M. Simchowitz, R. Tedrake, and D. Rus, "Faster algorithms for growing collision-free convex polytopes in robot configuration space," *arXiv preprint*, 2024.

[12] D. von Wrangel and R. Tedrake, "Using graphs of convex sets to guide nonconvex trajectory optimization," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2024.

[13] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polygons," in *Proceedings of the twentieth annual symposium on Computational geometry*, 2004, pp. 17–26.

[14] H. Liu, W. Liu, and L. J. Latecki, "Convex shape decomposition," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE, 2010, pp. 97–104.

[15] K. Mamou and F. Ghorbel, "A simple and efficient approach for 3d mesh approximate convex decomposition," in *2009 16th IEEE international conference on image processing (ICIP)*. IEEE, 2009, pp. 3501–3504.

[16] M. Stilman, "Global manipulation planning in robot joint space with task constraints," *IEEE Transactions on Robotics*, 2010.

[17] D. Berenson, *Constrained manipulation planning*. Carnegie Mellon University, 2011.

[18] N. T. Dantam, Z. K. Kingston, S. Chaudhuri, and L. E. Kavraki, "Incremental task and motion planning: A constraint-based approach." in *Robotics: Science and systems*, vol. 12. Ann Arbor, MI, USA, 2016, p. 00052.

[19] M. Bonilla, E. Farnioli, L. Pallottino, and A. Bicchi, "Sample-based motion planning for soft robot manipulators under task constraints," in *IEEE International Conference on Robotics and Automation*, 2015.

[20] S. Rodriguez, S. Thomas, R. Pearce, and N. M. Amato, "Resampl: A region-sensitive adaptive motion planner," in *Algorithmic foundation of robotics VII: selected contributions of the seventh international workshop on the algorithmic foundations of robotics*. Springer, 2008.

[21] L. Brudermüller, T. Lembono, S. Shetty, and S. Calinon, "Trajectory prediction with compressed 3d environment representation using tensor train decomposition," in *2021 20th International Conference on Advanced Robotics (ICAR)*. IEEE, 2021, pp. 633–639.

[22] I. V. Oseledets, "Tensor-train decomposition," *SIAM Journal on Scientific Computing*, vol. 33, no. 5, pp. 2295–2317, 2011.

[23] S. Dolgov and R. Scheichl, "A hybrid alternating least squares–tt-cross algorithm for parametric pdes," *SIAM/ASA Journal on Uncertainty Quantification*, vol. 7, no. 1, pp. 260–291, 2019. [Online]. Available: https://doi.org/10.1137/17M1138881

[24] I. Oseledets and E. Tyrtyshnikov, "Tt-cross approximation for multidimensional arrays," *Linear Algebra and its Applications*, vol. 432, no. 1, pp. 70–88, 2010. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0024379509003747

[25] T. Marcucci, M. Petersen, D. von Wrangel, and R. Tedrake, "Motion planning around obstacles with convex optimization," *Science robotics*, vol. 8, no. 84, p. eadf7843, 2023.

[26] T. Marcucci, J. Umenberger, P. Parrilo, and R. Tedrake, "Shortest paths in graphs of convex sets," *SIAM Journal on Optimization*, vol. 34, no. 1, pp. 507–532, 2024.

[27] T. Yoshikawa, "Manipulability of robotic mechanisms," *The international journal of Robotics Research*, vol. 4, no. 2, pp. 3–9, 1985.

[28] F. Marić, L. Petrović, M. Guberina, J. Kelly, and I. Petrović, "A riemannian metric for geometry-aware singularity avoidance by articulated robots," *Robotics and autonomous systems*, vol. 145, p. 103865, 2021.

[29] S. Shetty, T. Lembono, T. Loew, and S. Calinon, "Tensor train for global optimization problems in robotics," *The International Journal of Robotics Research*, vol. 43, no. 6, pp. 811–839, 2024.

[30] S. Dolgov, K. Anaya-Izquierdo, C. Fox, and R. Scheichl, "Approximation and sampling of multivariate probability distributions in the tensor train decomposition," *Statistics and Computing*, vol. 30, no. 3, pp. 603–625, 2020.

[31] A. Bryant and K. Cios, "Rnn-dbscan: A density-based clustering algorithm using reverse nearest neighbor density estimates," *IEEE Transactions on Knowledge and Data Engineering*, vol. 30, no. 6, pp. 1109–1121, 2017.

[32] C. L. Luck and S. Lee, "Self-motion topology for redundant manipulators with joint limits," in *[1993] Proceedings IEEE International Conference on Robotics and Automation*. IEEE, 1993, pp. 626–631.

[33] S. Haddadin, "The franka emika robot: A standard platform in robotics research," *IEEE Robotics & Automation Magazine*, 2024.

[34] A. Novikov, D. Podoprikhin, A. Osokin, and D. P. Vetrov, "Tensorizing neural networks," *Advances in neural information processing systems*, vol. 28, 2015.

[35] P. Tichavsky and O. Straka, "Optimizing the order of modes in tensor train decomposition," *IEEE Signal Processing Letters*, 2025.

[36] Q.-C. Pham, "A general, fast, and robust implementation of the time-optimal path parameterization algorithm," *IEEE Transactions on Robotics*, vol. 30, no. 6, pp. 1533–1540, 2014.

[37] H. Oleynikova, A. Millane, Z. Taylor, E. Galceran, J. Nieto, and R. Siegwart, "Signed distance fields: A natural representation for both mapping and planning," in *RSS 2016 workshop: geometry and beyond-representations, physics, and scene understanding for robotics*. University of Michigan, 2016.