2020 IEEE International Conference on Robotics and Automation (ICRA)
31 May - 31 August, 2020. Paris, France

Improved C-Space Exploration and Path Planning for Robotic Manipulators Using Distance Information

Bakir Lacevic and Dinko Osmankovic

***Abstract*— We present a simple method to quickly explore C-spaces of robotic manipulators and thus facilitate path planning. The method is based on a novel geometrical structure called *generalized bur*. It is a star-like tree, rooted at a given point in free C-space, with an arbitrary number of guaranteed collision-free edges computed using distance information from the workspace and simple forward kinematics. Generalized bur captures large portions of free C-space, enabling accelerated exploration. The workspace is assumed to be decomposable into a finite set of (possibly overlapping) convex obstacles. When plugged in a suitable RRT-like planning algorithm, generalized burs enable significant performance improvements, while at the same time enabling exact collision-free paths.**

I. INTRODUCTION

The original sampling-based algorithms [1], [2], [3], with a variety of follow-ups (e.g., [4], [5], [6]), typically use collision checking to validate local paths up to a given resolution. Free local paths are assembled into a graph capturing the connectivity of free C-space ($\mathcal{C}_f$) to find collision-free paths. More reliable path verification is based on methods such as continuous collision detection [7], [8], enlarged robot models [9], and distance queries [10], [11], [9], [8], [12]. An interesting concept of *tiling roadmaps* to facilitate planning for free-flying multi-link robots can be found in [13], while in [14] authors propose *rapidly exploring random vines* to tackle the problem of narrow passages.

Distance query has facilitated certain aspects of path planning. *Bubbles of free C-space* [11] are used in [10] to compute RRT extensions. In [15], the distance measure is used to grow a volumetric tree that biases the sampling process. The method from [16] uses distance information to decimate subsequent collision checks within a *safety certificates* framework. The recent paper [17] uses distance as a cost function within the roadmap-based planning in unknown environments. In [18], the structure called *bur of free C-space* is used within multi-directional RRT-like algorithm to accelerate the exploration of $\mathcal{C}_f$. The bur provides significantly larger portion of $\mathcal{C}_f$ when compared with the bubble, while using the same distance query. The proposed rapidly exploring bur tree (RBT) outperformed both classical RRT and related algorithms that use distance information.

This work reveals the underlying distance information to further expand the knowledge about $\mathcal{C}_f$. The following assumptions have been made: *i*) the robot belongs to a class of fixed-base manipulators with rotational joints, and *ii*) the environment is a set of (possibly overlapping) convex obstacles. In case the latter does not hold, a variety of methods are available for practical convex decomposition of meshes and polyhedra, e.g., [19], [20], [21]. The contribution highlights are as follows.

- *The definition of the generalized bur*—it is a significantly enlarged version of the original bur (Fig. 1) computable using the same amount of input information.
- *A plain path planning algorithm*—when plugged in a RRT-like algorithm, generalized burs provide significant performance improvement w.r.t. related methods (Fig. 2). The algorithm is simple in structure and requires only basic procedures: distance query, nearest neighbors and forward kinematics. No heavy preprocessing, learning nor complicated data structures are necessary.

The paper is organized as follows. Section II recollects the bubbles and burs of free C-space. The generalized bur is defined in Section III, while the planning algorithm is described in Section IV. Section V brings the validation study while the final remarks are given in Section VI.

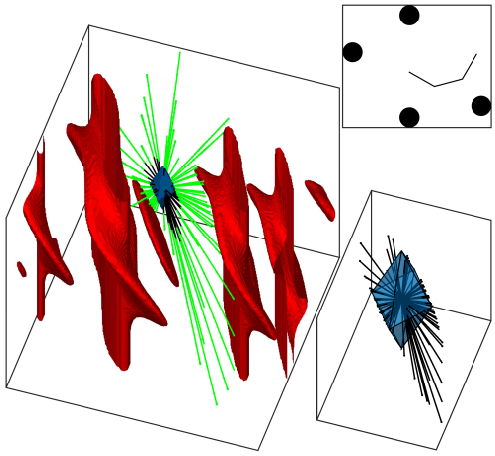

Fig. 1: C-space with obstacles for a 3 DOF planar robot. Three collision-free structures obtained using a single distance query: bubble of free C-space (blue) [11], bur of free C-space (black) [18], and a generalized bur (green). The workspace is depicted in the top-right, while the zoomed versions of bubble and bur are shown in the bottom right.

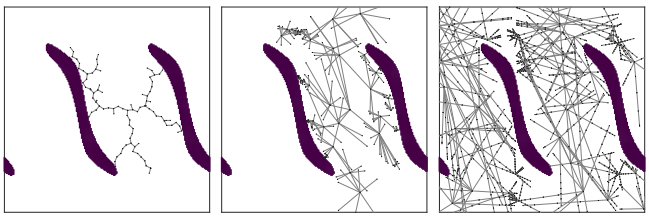

Fig. 2: C-space coverage after 50ms: simple RRT [2], bur tree [18], and proposed generalized bur tree (see Section IV)

Authors are with the Faculty of Electrical Engineering, University of Sarajevo, 71000 Sarajevo, Bosnia and Herzegovina. {bakir.lacevic,dinko.osmankovic}@etf.unsa.ba

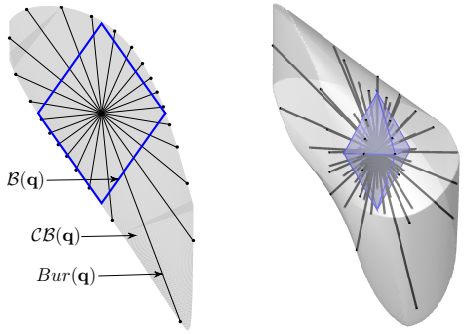

Fig. 3: Bubble, complete bubble and bur for 2 and 3 DOF planar robots

II. BACKGROUND - BUBBLES AND BURS

Bubbles of free C-space are proposed in [11]. The bubble $\mathcal{B}(\mathbf{q}, d_c)$ is the volume of free C-space around the configuration $\mathbf{q}$. Its computation requires the minimum workspace distance d_c between the robot and set of obstacles. For a manipulator with n revolute joints, it is defined as [11]:

$$\mathcal{B}(\mathbf{q}, d_c) = \left\{ \mathbf{y} \;\middle|\; \textstyle\sum_{i=1}^{n} r_i \left| y_i - q_i \right| \leq d_c \right\}. \tag{1}$$

The weight r_i is the radius of the cylinder coaxial with i-th joint. This cylinder has to enclose all the links starting from i-th joint to the end-effector.

The expression $\sum_{i=1}^{n} r_i \left| y_i - q_i \right|$ is a conservative upper bound on the displacement of any point on the manipulator when the configuration changes from $\mathbf{q} = (q_1 \dots q_n)^T$ to $\mathbf{y} = (y_1 \dots y_n)^T$. Changing the configuration from $\mathbf{q}$ to an arbitrary configuration $\mathbf{y}$ within a bubble means that no point on the robot moves more than d_c, which guarantees that no collision occurs. From (1), it follows that bubbles are diamond-shaped (see Fig. 3 for a 2D case). It was hinted in [11] that the computation of bubbles using (1) may be unnecessarily conservative. In that regard, the method to construct larger local volumes of free C-space based on the same distance information was proposed in [18]. Firstly, the *complete bubble of free C-space* is defined [18]:

Definition 1: Let $\mathbf{q} \in \mathcal{C}_f$, and let d_c be the minimum distance of the robot $\mathcal{R}$ at the configuration $\mathbf{q}$ to a set of workspace obstacles. The complete bubble of free C-space, w.r.t. configuration $\mathbf{q}$, is the set:

$$\mathcal{CB}(\mathbf{q}, d_c) = \left\{ \mathbf{x} \in \mathcal{C}_f \;\middle|\; \rho_{\mathcal{R}}(\mathbf{q}, \mathbf{y}) < d_c,\ \forall \mathbf{y} \in \overline{\mathbf{qx}} \right\}, \tag{2}$$

where $\overline{\mathbf{qx}}$ is the line segment with endpoints $\mathbf{q}$ and $\mathbf{x}$. The metric $\rho_{\mathcal{R}}(\mathbf{q}_1, \mathbf{q}_2)$ used above is defined as:

$$\rho_{\mathcal{R}}(\mathbf{q}_1, \mathbf{q}_2) = \max_{\mathbf{p} \in \mathcal{R}} \left\| \mathbf{f}_{\mathbf{p}}(\mathbf{q}_1) - \mathbf{f}_{\mathbf{p}}(\mathbf{q}_2) \right\|, \tag{3}$$

where $\mathbf{f}_{\mathbf{p}}(\mathbf{z})$ stands for the Cartesian coordinates of the point $\mathbf{p}$, on the robot $\mathcal{R}$, which is at the configuration $\mathbf{z} \in \mathcal{C}$.

The complete bubble $\mathcal{CB}(\mathbf{q}, d_c)$ is a superset of the original bubble $\mathcal{B}(\mathbf{q}, d_c)$, since the weighted L_1 metric used in (1) is more conservative than $\rho_{\mathcal{R}}$. Fig. 3 depicts a bubble and a complete bubble for 2 DOF and 3 DOF planar robots. The exact computation of the complete bubble is hard and practically impossible for robots with more than 2 DOFs [18]. For instance, the complete bubble for 3 DOF robot in Fig. 3 is obtained via exhaustive numerical computation. Moreover, the feature of convexity is lost in the general case. However, it turns out that the border of the complete bubble can be sampled efficiently with the guarantee that line segments, which join $\mathbf{q}$ with the samples on the border, are collision-free. This follows from the fact that the complete bubble is star-convex by definition. The resulting structure is denoted as the bur of free C-space (see Fig. 3). The formal definition of the bur is [18]:

Definition 2: Let $Q_e = \{\mathbf{q}_{e_1}, \dots, \mathbf{q}_{e_N}\}$ be the set of N points in C-space such that $\mathbf{q}_{e_i} \neq \mathbf{q}$. The bur of free C-space at the configuration $\mathbf{q}$, w.r.t. Q_e is given by:

$$Bur\left(\mathbf{q}, Q_e, d_c\right) = \mathcal{CB}(\mathbf{q}, d_c) \cap \left\{ \overline{\mathbf{qq}}_{e_1}, \dots, \overline{\mathbf{qq}}_{e_N} \right\}. \tag{4}$$

Here, Q_e is the set of N points (usually random) in C-space that are far enough from the configuration $\mathbf{q}$, i.e., do not belong to the complete bubble $\mathcal{CB}(\mathbf{q}, d_c)$. This is achieved by setting a large value of δ (see Algorithm 2 in Section IV).

The computation of a bur can be decomposed into computation of individual spines (rays). It reduces to finding the second endpoint of the spine, which is further equivalent with the solution to a following problem. Given a current robot configuration $\mathbf{q}$, a distance d_c, and a remote configuration $\mathbf{q}_e$, move as far as possible along the line segment $\overline{\mathbf{qq}_e}$, starting from $\mathbf{q}$, until some point on the robot gets displaced exactly by d_c. If this does not occur before reaching $\mathbf{q}_e$, then the configuration $\mathbf{q}_e$ can be declared as the second endpoint of the spine. This approach may resemble the exact collision checking proposed in [22], [23] and recently revisited in [24]. However, said methods validate the existing candidate for a local path, whereas we aim at local path construction without invoking distance query more than once. Formally, the second endpoint of the spine $\mathbf{q}^*$ can be defined as:

$$\mathbf{q}^*(\mathbf{q}, \mathbf{q}_e) = \underset{\substack{\mathbf{y} \in \overline{\mathbf{qq}_e} \\ \rho_{\mathcal{R}}(\mathbf{z}, \mathbf{q}) < d_c, \forall \mathbf{z} \in \overline{\mathbf{qy}}}}{\arg\max} \left\| \mathbf{y} - \mathbf{q} \right\|. \tag{5}$$

The metric ρ can be reformulated as [18]:

$$\rho_{\mathcal{R}}(\mathbf{q}_1, \mathbf{q}_2) = \max_i \left\| \mathbf{f}_{\mathbf{p}_i}(\mathbf{q}_1) - \mathbf{f}_{\mathbf{p}_i}(\mathbf{q}_2) \right\|, \tag{6}$$

where $\mathbf{f}_{\mathbf{p}_i}(\mathbf{z})$ denotes the workspace Cartesian coordinates of the i-th link's distal point. Thus, the metric $\rho_{\mathcal{R}}$ can be obtained by simply applying forward kinematics. Observing the parameterization of the line segment $\overline{\mathbf{qq}_e}$ given by

$$\mathbf{q}(t) = \mathbf{q} + t\left(\mathbf{q}_e - \mathbf{q}\right), \quad t \in [0, 1], \tag{7}$$

it has been shown in [18] that the value of t^*, which corresponds to the endpoint of the spine $\mathbf{q}^*$ can be obtained as the limit of the following sequence:

$$t_{k+1} = t_k + \frac{\varphi(t_k)}{\sum_{i=1}^{n} r_i(t_k) \left| q_{ei} - q_{ki} \right|} (1 - t_k), \tag{8}$$

where $t_0 = 0$, $r_i(t_k)$ are the radii of enclosing cylinders when the robot is at the configuration $\mathbf{q}_k = \mathbf{q}(t_k)$ and

$$\varphi(t) = d_c - \rho_{\mathcal{R}}\left(\mathbf{q}, \mathbf{q} + t(\mathbf{q}_e - \mathbf{q})\right). \tag{9}$$

It turns out that the sequence (8) converges fast and it usually takes less than five iterations to reach a close approximation of t^*, i.e., $\mathbf{q}^* = \mathbf{q}(t^*)$. Note that all the ingredients necessary

to conduct a single iteration (radii $r_i(t_k)$ and the function $\varphi(t_k)$) are easily obtained by performing a single forward kinematics computation at the configuration $\mathbf{q}_k = \mathbf{q}(t_k)$.

III. BEYOND COMPLETE BUBBLES AND BURS

A natural question arises whether the information about the free C-space can be enriched in terms of expanding the limits of a complete bubble (or a bur). An intuitive approach is to propagate the existing complete bubble by computing new complete bubbles around the configuration points that belong to the border of the original complete bubble. Thus, an enlarged volume of free C-space would be obtained. The border of such enlarged structure can be further propagated in a similar fashion. Clearly, such structure would not bring much practical significance for the following reasons. Firstly, the original complete bubble is extremely hard to compute, even for simple kinematic chains with few degrees of freedom. Secondly, even if it was available, the further propagation of free C-space would require an infinite number of distance queries.

On the other hand, the structure obtained by the described thought experiment inherits the star-convexity feature from the complete bubble. Hence, it calls for an attempt to perform a discrete radial representation, in a similar fashion as the definition of bur was motivated by the aim of reaching the border of the complete bubble (see Definition 2). To this end, we define the structure called *the generalized bur*.

Definition 3: Let $Q_e = \{\mathbf{q}_{e_1}, \ldots, \mathbf{q}_{e_N}\}$ be the set of N points in C-space such that $\mathbf{q}_{e_i} \neq \mathbf{q}$. The generalized bur (of order $k+1$) is given by the following recursive sequence:

$$\begin{aligned} &GBur_0(\mathbf{q}, Q_e, d(\mathbf{q})) = Bur(\mathbf{q}, Q_e, d(\mathbf{q})), \\ &GBur_{k+1}(\mathbf{q}, Q_e, d(\mathbf{q})) = GBur_k(\mathbf{q}, Q_e, d(\mathbf{q})) \\ &\bigcup \left\{ \bigcup_{i=1}^{N} [\overline{\mathbf{y}_{i,k}\mathbf{q}_{ei}} \bigcap \mathcal{CB}(\mathbf{y}_{i,k}, d(\mathbf{y}_{i,k}))] \right\}, \end{aligned} \tag{10}$$

where $k \in \mathbb{N} \cup \{0\}$ and configurations $\mathbf{y}_{1,k}, \mathbf{y}_{2,k}, \ldots, \mathbf{y}_{N,k}$ denote the endpoints of $GBur_k(\mathbf{q}, Q_e, d(\mathbf{q}))$.

In other words, to obtain the generalized bur $GBur_1(\mathbf{q}, Q_e, d(\mathbf{q}))$, each spine $\overline{\mathbf{q}\mathbf{y}_{j,0}}$, $j = 1, 2, \ldots, N$, of the starting bur $Bur(\mathbf{q}, Q_e, d(\mathbf{q}))$ is further extended from its endpoint $\mathbf{y}_{j,0}$ in the same direction until it reaches the border of the complete bubble defined around the said endpoint. By definitions of bur and complete bubble, this extra length equals the corresponding spine of the bur computed at $\mathbf{y}_{j,0}$. The process can be repeated for each spine and for arbitrarily many times.

Though generalized bur $GBur_k(\mathbf{q}, Q_e, d(\mathbf{q}))$ can be computed for an arbitrary $k \in \mathbb{N}$, it may still be of questionable practical significance. Namely, to compute $GBur_k(\mathbf{q}, Q_e, d(\mathbf{q}))$, additional kN distance queries need to be performed, which may turn out to be too expensive.

In this regard, we design the procedure that substantially alleviates the computational cost of obtaining generalized burs. This procedure is based on replacing the distance term $d(\mathbf{y}_{i,k})$ from (10) with a useful underestimation that can be obtained using only simple algebra and, most importantly, without invoking actual distance query. This may result in a slight loss of generalized bur size. However, the resulting "trimmed" structure (denoted in the sequel by $\underline{GBur}_k(\mathbf{q}, Q_e)$) is in general still substantially larger than the starting bur. The bottom line is that the $\underline{GBur}_k(\mathbf{q}, Q_e)$ of an arbitrary order k and with an arbitrary number of spines (i.e., the cardinality of Q_e) can be obtained using only a single distance query (necessary to compute the starting bur) and a limited number of forward kinematics computations.

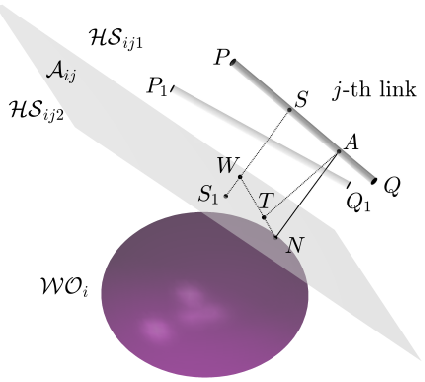

Fig. 4: Line segment $\overline{PQ}$ and a world obstacle $\mathcal{WO}_i$

In the sequel, we describe the procedure of computing the underestimation $\underline{d}(\mathbf{y}_{i,k})$ of the term $d(\mathbf{y}_{i,k})$. We assume the environment is represented by a set of m convex obstacles $\mathcal{WO} = \{\mathcal{WO}_1, \ldots, \mathcal{WO}_m\}$, and we assume the distance query returns mn distances w.r.t. n robot links and m obstacles. It may appear that thus we actually have mn distance queries instead of a single one. However, it turns out that the time necessary to obtain these mn distances is only marginally larger than the time required to compute a single distance between the robot and the set of obstacles (in case both robot and the set of obstacles are represented by single collision objects - see details of RBT-CONNECT implementation in section V).

We first focus on the relationship between a single link and a single obstacle (Fig. 4). For now, the link is observed as a line segment, though the approach can easily tackle more complex geometries. Let d_{ij} be the minimum distance between the robot's j-th link denoted by $\overline{PQ}$ and the obstacle $\mathcal{WO}_i$. Let N and A be the pair of closest points that belong to the obstacle and link respectively. If such pair is not unique, we pick an arbitrary one. The plane $\mathcal{A}_{ij}$ at point N is perpendicular to the segment $\overline{NA}$. This plane separates the workspace into two halfspaces: $\mathcal{HS}_{ij1}$ and $\mathcal{HS}_{ij2}$. The following theorem (see Appendix for a proof) provides some useful insight into relative position between the link $\overline{PQ}$, the plane $\mathcal{A}_{ij}$ and the convex obstacle $\mathcal{WO}_i$.

Theorem 1: Let S be the arbitrary point on the link $\overline{PQ}$. Then, the following holds:

$$\rho(S, \mathcal{WO}_i) \geq \rho(S, \mathcal{A}_{ij}) \geq d_{ij}, \tag{11}$$

where $\rho(X, Y)$ is the minimum distance between X and Y. This implies that the plane $\mathcal{A}_{ij}$ separates the link $\overline{PQ}$ from the obstacle $\mathcal{WO}_i$ with $\overline{PQ}$ belonging to halfspace $\mathcal{HS}_{ij1}$ with the clearance margin d_{ij}. On the other hand, the obstacle $\mathcal{WO}_i$ belongs to halfspace $\mathcal{HS}_{ij2}$.

Assume that the position of the link $\overline{PQ}$ is implied by the configuration $\mathbf{q}$. Then, $Bur(\mathbf{q}, Q_e, d_c)$ can be computed where $d_c \leq d_{ij}$ is the minimum distance robot-obstacles. Let $\mathbf{y}_{l,0}$ be the endpoint of l-th spine ($l \in \{1, \ldots, N\}$) of the bur $Bur(\mathbf{q}, Q_e, d_c) = GBur_0(\mathbf{q}, Q_e, d_c)$. If the robot moves from $\mathbf{q}$ to $\mathbf{y}_{l,0}$ along the spine $\overline{\mathbf{q}\mathbf{y}_{l,0}}$, the link $\overline{PQ}$ will be relocated to another position $\overline{P_1Q_1}$ (Fig. 4). Based on proven bur properties [18], an endpoint (e.g., point Q) of the link $\overline{PQ}$ will be moved from its starting position by the distance $d_Q \leq d_c \leq d_{ij}$, while all other link points are moved by distance that is less or equal d_Q. According to Theorem 1, the link $\overline{P_1Q_1}$ remains in the halfspace $\mathcal{HS}_{ij1}$, and it holds that $\rho\left(\overline{P_1Q_1}, \mathcal{WO}_i\right) \geq \rho\left(\overline{P_1Q_1}, \mathcal{A}_{ij}\right) \geq 0$. Thus, instead of invoking the distance query to compute $\rho\left(\overline{P_1Q_1}, \mathcal{WO}_i\right)$, one may use the underestimation:

$$\underline{d}_{ij}\left(\mathbf{y}_{l,0}\right) \equiv \rho\left(\overline{P_1Q_1}, \mathcal{A}_{ij}\right) = \min_{X \in \{P_1, Q_1\}} \rho\left(X, \mathcal{A}_{ij}\right), \quad (12)$$

which reduces to trivial algebra: point-to-plane distance computation. This result can be generalized to tackle at least two more complex link geometries. The first case is the capsule–a Minkowski sum of the segment and a sphere. The capsule may be observed as a conservative cover of the real link. In this case, the distance from plane is obtained simply by subtracting the sphere radius from the result computed by (12). Alternative representation is the convex hull of the real link. The distance is then estimated in a similar way as in (12), with the minimization performed over all the hull's vertices. The simplest, yet the most conservative approach is to use the oriented bounding box (OBB) of the link.

Finally, in order to continue the extension of the l-th spine of $Bur(\mathbf{q}, Q_e, d_c) = GBur_0(\mathbf{q}, Q_e, d_c)$ towards $\mathbf{q}_{el}$, the underestimation $\underline{d}\left(\mathbf{y}_{l,0}\right)$ of the term $d\left(\mathbf{y}_{l,0}\right)$ from (10) is chosen as the minimum value of $\underline{d}_{ij}\left(\mathbf{y}_{l,0}\right)$ over all the obstacles $\mathcal{WO}_i$ (or more precisely planes $\mathcal{A}_{ij}$), $i = 1, 2, \ldots, m$ and all the links $j = 1, 2, \ldots, n$, i.e.,

$$\underline{d}\left(\mathbf{y}_{l,0}\right) = \min_j \left\{\min_i \left\{\underline{d}_{ij}\left(\mathbf{y}_{l,0}\right)\right\}\right\}. \quad (13)$$

Once we have $\underline{d}\left(\mathbf{y}_{l,0}\right)$, the spine $\overline{\mathbf{q}\mathbf{y}_{l,0}}$ can be extended towards $\mathbf{q}_{el}$ by the length $\overline{\mathbf{y}_{l,0}\mathbf{y}_{l,1}} = \overline{\mathbf{y}_{l,0}\mathbf{q}_{el}} \bigcap \mathcal{CB}\left(\mathbf{y}_{l,0}, \underline{d}\left(\mathbf{y}_{l,0}\right)\right)$, thus adding one of N extensions in the process of constructing $\underline{GBur}_1(\mathbf{q}, Q_e)$. Analogously, the remaining $N-1$ extensions can be computed to complete the $\underline{GBur}_1(\mathbf{q}, Q_e)$. Clearly, $\underline{GBur}_1(\mathbf{q}, Q_e)$ may be further propagated to $\underline{GBur}_2(\mathbf{q}, Q_e)$ using the same procedure. For instance, when the robot moves from configuration $\mathbf{y}_{l,0}$ to $\mathbf{y}_{l,1}$, the j-th link gets relocated from $\overline{P_1Q_1}$ to $\overline{P_2Q_2}$ that still belongs to halfspace $\mathcal{HS}_{ij1}$. Again, the "fake" distance query may compute the distance from $\overline{P_2Q_2}$ to $\mathcal{A}_{ij}$. The result is then used to extend from $\mathbf{y}_{l,1}$ to $\mathbf{y}_{l,2}$. The process is repeated in the same fashion for other endpoints and arbitrarily many times k to compute $\underline{GBur}_k(\mathbf{q}, Q_e)$. The function GENERALIZED_BUR shows the pseudocode of the algorithm for computing the generalized bur of order k using distance underestimations. Fig. 5 shows several examples of generalized bur in 2D C-space computed by the proposed method with clear indication of substantially larger area of free C-space covered.

Algorithm 1 GENERALIZED_BUR($\mathbf{q}, Q_e, k$)

1: $[d_{ij}, \mathcal{A}_{ij}] \leftarrow$ DISTANCE_QUERY($\mathbf{q}, \mathcal{WO}$);
2: $d_c \leftarrow \min_{i,j} d_{i,j}$;
3: $\underline{GBur} \leftarrow Bur(\mathbf{q}, Q_e, dc)$; ▷ *Initialize $\underline{GBur}$ to ordinary Bur*
4: **for** $l = 1$ **to** N **do** ▷ *All directions—spines*
5: **for** $p = 0$ **to** $k-1$ **do** ▷ *All extensions in one direction*
6: $\mathbf{y}_{l,p} \leftarrow$ ENDPOINT($\underline{GBur}, l$);
7: $\underline{d} \leftarrow \min_j \left\{\min_i \left\{\rho\left(\text{LINK}_j(\mathbf{y}_{l,p}), \mathcal{A}_{ij}\right)\right\}\right\}$;
8: $\overline{\mathbf{y}_{l,p}\mathbf{y}_{l,p+1}} = \overline{\mathbf{y}_{l,p}\mathbf{q}_{el}} \bigcap \mathcal{CB}\left(\mathbf{y}_{l,p}, \underline{d}\right)$;
9: $\underline{GBur} \leftarrow \underline{GBur} \bigcup \overline{\mathbf{y}_{l,p}\mathbf{y}_{l,p+1}}$;
10: **return** $\underline{GBur}$

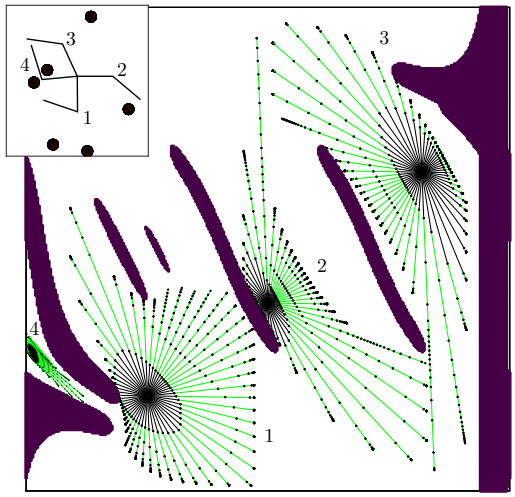

Fig. 5: Generalized burs of order 20 that correspond to four different configurations of 2 DOF planar manipulator. The starting bur $GBur_0(\mathbf{q}, Q_e) = Bur(\mathbf{q}, Q_e)$ is colored black while the propagation to $\underline{GBur}_k(\mathbf{q}, Q_e)$, $k = 1, \ldots, 20$ is colored green. The workspace is depicted in the upper left corner.

IV. PATH PLANNING ALGORITHM

The algorithm is a straightforward modification of RRT. Instead of performing a single extension from the nearest neighbor $\mathbf{q}_{near}$ toward the randomly picked configuration $\mathbf{q}_{rand}$, we do a multidirectional extension by growing a generalized bur $GBur(\mathbf{q}_{near})$ from $\mathbf{q}_{near}$. Voronoi bias is preserved by imposing that one spine of $GBur(\mathbf{q}_{near})$ has to be directed toward $\mathbf{q}_{rand}$. We refer to the resulting structure/algorithm as to rapidly exploring generalized bur tree (RGBT). Again, the bidirectional version inspired by [3] is investigated. Functions RGBT-CONNECT and GBUR_CONNECT show the pseudocode of the algorithm. As in [18], the algorithm may enter the RRT mode, if the workspace distance drops below a given threshold (lines 5-10 in RGBT-CONNECT and lines 9-15 in GBUR_CONNECT). Fig. 2 shows the exploratory potential of the unidirectional RGBT compared to those of RBT [18] and RRT [2]. RGBT clearly covers the free C-space much faster.

V. SIMULATION STUDY

Numerical experiments are implemented in C++, where distance/collision queries have been performed using FCL library [25]. The nearest neighbor query uses FLANN library

Algorithm 2 RGBT-CONNECT($\mathbf{q}_{start}, \mathbf{q}_{goal}$)

1: $\mathcal{T}_a \leftarrow \mathbf{q}_{start}$, $\mathcal{T}_b \leftarrow \mathbf{q}_{goal}$
2: **for** $i = 1$ **to** i_{max} **do**
3: $\mathbf{y}_{e_1} \leftarrow$ RANDOM_CONFIG()
4: $\mathbf{q}_{near} \leftarrow$ NEAREST($\mathbf{y}_{e_1}, \mathcal{T}_a$)
5: **if** $d_c(\mathbf{q}_{near}) < d_k$ **then** ▷ *RRT mode*
6: $\mathbf{q}_{new} \leftarrow \mathbf{q}_{near} + \varepsilon \frac{\mathbf{y}_{e_1} - \mathbf{q}_{near}}{\|\mathbf{y}_{e_1} - \mathbf{q}_{near}\|}$
7: **if** $\neg Collision(\mathbf{q}_{new}, \mathbf{q}_{near})$ **then**
8: $\mathcal{T}_a \leftarrow \mathcal{T}_a \cup \overline{\mathbf{q}_{new}\mathbf{q}_{near}}$
9: **else**
10: $Continue$
11: **else**
12: $\mathbf{q}_{e_1} \leftarrow \mathbf{q}_{near} + \delta \frac{\mathbf{y}_{e_1} - \mathbf{q}_{near}}{\|\mathbf{y}_{e_1} - \mathbf{q}_{near}\|}$ ▷ *The first spine points*
13: $Q_e \leftarrow \mathbf{q}_{e_1}$ ▷ *at the first random configuration* $\mathbf{y}_{e_1}$ ▷ *(distanced to* $\mathbf{q}_{e_1}$*)*
14: **for** $j = 2$ **to** N **do**
15: $\mathbf{y}_{e_j} \leftarrow$ RANDOM_CONFIG()
16: $\mathbf{q}_{e_j} \leftarrow \mathbf{q}_{near} + \delta \frac{\mathbf{y}_{e_j} - \mathbf{q}_{near}}{\|\mathbf{y}_{e_j} - \mathbf{q}_{near}\|}$
17: $Q_e \leftarrow Q_e \cup \mathbf{q}_{e_i}$ ▷ *Other random directions*
18: $\mathcal{T}_a \leftarrow \mathcal{T}_a \cup \underline{GBur}_k(\mathbf{q}_{near}, Q_e)$
19: $\mathbf{q}_{new} \leftarrow$ENDPOINT($\underline{GBur}_k(\mathbf{q}_{near}, Q_e), \mathbf{q}_{e_1}$)
▷ *Endpoint of the first spine as a target for connect heuristic*
20: **if** GBUR_CONNECT($\mathcal{T}_b, \mathbf{q}_{new}$) $= Reached$ **then**
21: **return** PATH($\mathcal{T}_a, \mathcal{T}_b$)
22: SWAP($\mathcal{T}_a, \mathcal{T}_b$)
23: **return** $Failure$

Algorithm 3 GBUR_CONNECT($\mathcal{T}, \mathbf{q}$)

1: $\mathbf{q}_n \leftarrow$ NEAREST($\mathbf{q}, \mathcal{T}$), $\mathbf{q}_0 \leftarrow \mathbf{q}_n$
2: **repeat**
3: **if** $d_c(\mathbf{q}_n) > d_k$ **then**
4: $\mathbf{q}_t \leftarrow$ ENDPOINT($\underline{GBur}_k(\mathbf{q}_n, \mathbf{q}), \mathbf{q}$) ▷ *Endpoint of single-spine GBur directing at* $\mathbf{q}$
5: $\Delta s \leftarrow \|\mathbf{q}_t - \mathbf{q}_n\|$
6: $\mathbf{q}_n \leftarrow \mathbf{q}_t$
7: **if** $\mathbf{q}_n = \mathbf{q}$ **then**
8: **return** $Reached$
9: **else** ▷ *RRT mode*
10: $\mathbf{q}_t \leftarrow \mathbf{q}_n + \varepsilon \frac{\mathbf{q} - \mathbf{q}_n}{\|\mathbf{q} - \mathbf{q}_n\|}$
11: **if** $\neg Collision(\mathbf{q}_t, \mathbf{q}_n)$ **then**
12: $\mathbf{q}_n \leftarrow \mathbf{q}_t$
13: **else**
14: **return** $Trapped$
15: **if** $\|\mathbf{q}_n - \mathbf{q}_0\| \geq \|\mathbf{q} - \mathbf{q}_0\|$ **then**
16: **return** $Reached$
17: **until** $\Delta s < Threshold$
18: **return** $Trapped$

with kd-tree approach [26]. All the simulations are performed on a PC with Intel Core i7-4702MQ 2.2 GHz and 16GB RAM. We compare performances of RBT-CONNECT from [18] and the proposed RGBT-CONNECT. For brevity, we omit RRT-CONNECT [3] from comparison since it was shown to be outperformed by RBT-CONNECT in [18]. Moreover, other related distance-based planners such as [9], [10] are omitted from this study mostly because they are either outperformed by RBT, as shown in [18] (such as the algorithm from [10]), or by RRT itself (self-reported in [9]).

Three robot types are considered: 2 and 8 DOF planar manipulators, and a 6 DOF ABB IRB120 industrial robot. Each robot is subjected to two scenarios (Fig. 6). The number of spines used for RBT expansion was set to $N = 7$, and kept constant during the algorithm run. This value was suggested in [18] for the fastest execution based on empirical analysis. It came as the trade-off between exploration potential (large N) and preventing the "explosion" in number of nodes to facilitate the nearest neighbor queries. We have not performed the dedicated analysis for RGBT regarding the optimal N, but rather kept the same value $N = 7$. An incremental performance improvement of RBT is possible by adaptive N during the algorithm (see [27]), but this option is out of this paper's scope. For each scene having more than one convex obstacle (2 DOF b), 8DOF b) and ABB b)), RBT-CONNECT [18] is tested in two modes. The first mode treats the environment as a single collision object, whereas the second mode observes the environment through multiple collision objects as an attempt to benefit from an already decomposed environment. It turns out that the second mode is consistently slower by 10-20%. Thus, we report only results for the first mode. For the computation of link-to-plane distances $\underline{d}_{ij}$ within RGBT-CONNECT, we used OBBs that cover corresponding links. Hence, the distance $\underline{d}_{ij}$ can be computed using (12) where the minimization is performed over eight OBB vertices. The nearest neighbor query (line 4, Alg. 2) observes the tree $\mathcal{T}_a$ as a set of nodes, where each generalized bur added to the tree (line 18, Alg. 2) contributes with all its nodes (up to kN nodes - see Algorithm 1). The RRT step size ε (line 6, Alg. 2) and parameter $Threshold$ (line 17, Alg. 3) are set to $3\frac{\pi}{180}$, while d_k (line 5, Alg. 2) is set to 0.005. The parameter δ (lines 12 and 16, Alg. 2) used for generating the set of remote points Q_e is set to 2π.

Fig. 6 shows the numerical performance indicators for the tested algorithms. Both planners achieved a 100% success rate for each scenario, where the results have been averaged over 100 algorithm runs. Mean values are shown in black while standard deviations are in gray. Beside intuitive features in Table I, the variable "#Extensions" denotes the number of additional extensions in generalized burs, i.e., the number of executions of lines 6-9 in Alg. 1. Note that the corresponding for-loop may be terminated should the extension $\overline{\mathbf{y}_{l,p}\mathbf{y}_{l,p+1}}$ drop below a given threshold (not shown in the pseudocode). RGBT planner consistently outperforms the RBT algorithm in all scenarios, with the runtime reduction from cca 50% (ABB b)) to 87% (2 DOF b)). Such improvement is attributed to substantially less iterations necessary to obtain the solution path, i.e., less full distance queries.

To test the limits of RGBT, we have conducted additional experiments where a single concave obstacle (a genus-two torus–Fig. 7) is subjected to several decomposition levels resulting in different cardinalities m of $\mathcal{WO} = \{\mathcal{WO}_1, \ldots, \mathcal{WO}_m\}$. To make sure comparison with RBT is fair within a given scenario, a decomposed obstacle is merged back into a single collision object. Fig. 8 shows the bar chart of average runtimes (over 100 runs) for two algorithms considering various levels of cardinality m. Again, the RGBT algorithm ouperforms RBT for each scenario. Up to $m \approx 100$, it is more than twice faster. For $m \approx 200$ the time reduction is cca 35%. We conjecture that further increase of m would cause the equalization of performances. This issue is left for future work since the current implementation cannot handle more detailed segmentation. On the other hand, we believe that the prospective use of inherently convex BVH-based representation of objects within FCL

Algorithm	Time (s)	Nodes	Iterations	#Extensions	Time (s)	Nodes	Iterations	#Extensions
	2 DOF a)				**2 DOF b)**			
RBT-CONNECT	0.304± 0.290	3317± 3055	208.8± 187.8	0± 0	11.274± 8.655	40025± 31201	6879± 4981	0± 0
RGBT-CONNECT	0.100± 0.075	1197± 822.6	18.7± 13.4	5.17± 0.49	1.450± 1.969	3144± 8581	1342± 1665	0.89± 0.46
	8 DOF a)				**8 DOF b)**			
RBT-CONNECT	1.887± 1.585	8151± 6742	499.6± 408.9	0± 0	9.315± 11.344	5988± 1375	3803± 3752	0± 0
RGBT-CONNECT	0.336± 0.100	1512± 461	29.6± 8.2	3.46± 0.44	3.251± 2.022	1566± 1100	1260± 681	0.28± 0.16
	ABB IRB120 a)				**ABB IRB120 b)**			
RBT-CONNECT	0.126± 0.090	72.82± 19.42	25.93± 17.69	0± 0	9.803± 5.304	3812± 3368	1847± 1020	0± 0
RGBT-CONNECT	0.041± 0.045	59.67± 13.07	18.44± 16.33	2.31± 0.09	4.865± 3.974	963± 342	1170± 880	0.30± 0.07

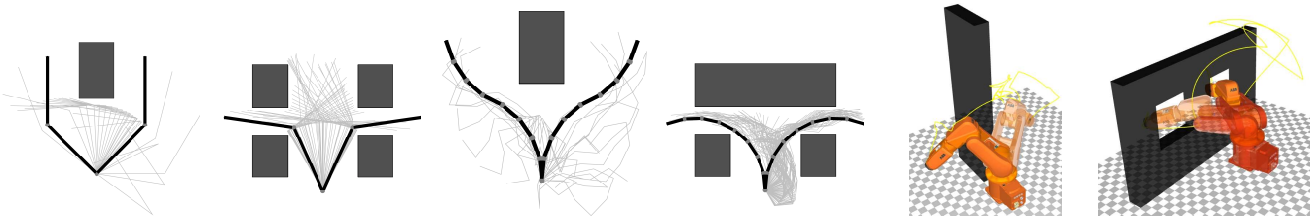

Fig. 6: Top - Numerical results. Bottom: scenarios for 3 robot types: 2 DOF a), 2DOF b), 8 DOF a), 8DOF b), IRB120 a), IRB120 b).

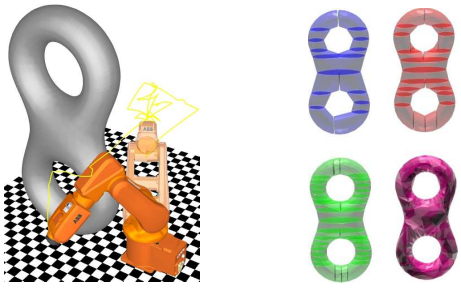

Fig. 7: Left - scenario with the concave obstacle. Right - various levels of convex decomposition.

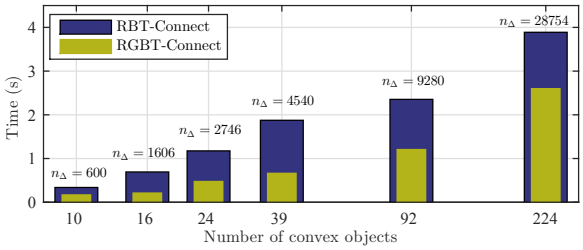

Fig. 8: Bar chart of average runtimes. The number of triangles n_Δ is indicated for each scenario.

would add to algorithm's performance and broaden its scope toward general environments.

VI. CONCLUSIONS

An approach to improve the C-space exploration of manipulators is presented. It is based on the structure called generalized bur that attempts to further exploit the workspace distance information, this time with some underlying, yet available knowledge about the geometry of both robot and the environment. This allows for substantial enhancement of exploratory capacities, utilized within a plain RRT-like planner which computes exact collision-free paths. Simulation study has shown that a novel algorithm consistently outperformed its predecessor and other related methods by transitivity. Some limitations are indicated as well.

Future work will address asymptotically optimal planning and a more flexible implementation within OMPL [28].

APPENDIX - *Proof of Theorem 1*

Proof: We first prove the inequality $\rho(S, \mathcal{A}_{ij}) \geq d_{ij}$. If the closest point A of the link $\overline{PQ}$ w.r.t. obstacle $\mathcal{WO}_i$ is interior point (as in Fig. 4), the vector $\overrightarrow{NA}$ is perpendicular to $\overline{PQ}$. Since $\overrightarrow{NA}$ is perpendicular $\mathcal{A}_{ij}$, it implies that $\overline{PQ}$ is parallel to $\mathcal{A}_{ij}$. Thus, $\rho(S, \mathcal{A}_{ij}) = d_{ij}$, $\forall S \in \overline{PQ}$. If the closest point A of the link $\overline{PQ}$ w.r.t. $\mathcal{WO}_i$ is the endpoint of $\overline{PQ}$, e.g., $A \equiv Q$, then $\angle NAP = \angle NQP \geq \frac{\pi}{2}$. Let $S \in \overline{PQ}$, $S \neq A$ and let the point S_1 be the orthogonal projection of S onto $\mathcal{A}_{ij}$. By Pythagorean theorem it holds that $\rho^2(S, \mathcal{A}_{ij}) = \overline{SS_1}^2 = \overline{NS}^2 - \overline{NS_1}^2$. Furthermore, since the $\angle NQP \geq \frac{\pi}{2}$, it follows that $\overline{NS}^2 \geq \overline{NA}^2 + \overline{AS}^2 = d_{ij} + \overline{AS}^2$. Since $\overline{NS_1}$ is the projection of $\overline{AS}$ onto $\mathcal{A}_{ij}$, we have that $\overline{AS} \geq \overline{NS_1}$. It follows that $\rho^2(S, \mathcal{A}_{ij}) \geq d_{ij}^2 + \overline{AS}^2 - \overline{NS_1}^2 \geq d_{ij}^2$. Now we prove that $\rho(S, \mathcal{WO}_i) \geq \rho(S, \mathcal{A}_{ij})$. Let $S \in \overline{PQ}$, $S \neq A$ and let the point S_1 be the orthogonal projection of S onto the plane $\mathcal{A}_{ij}$. Then $\rho(S, \mathcal{A}_{ij}) = \overline{SS_1}$. It is sufficient to show that no point $W \in \mathcal{WO}_i$ can belong to segment $\overline{SS_1}$. Assume the opposite, i.e., $\exists W \in \mathcal{WO}_i$ such that $W \in \overline{SS_1}$. This means that $\exists W \in \mathcal{HS}_{ij1}$ and the scalar product $\overrightarrow{NW} \cdot \overrightarrow{NA} > 0$. This can be written as: $\left(\overrightarrow{NA} + \overrightarrow{AW}\right)\overrightarrow{NA} > 0 \Rightarrow d_{i,j}^2 > \overrightarrow{AW}\overrightarrow{AN}$. Putting $x \equiv \overline{AW}$ and $\delta \equiv \angle NAW$, we have that $x \cos\delta < d_{ij}$. Now, let T be the arbitrary point on the line segment $\overrightarrow{NW}$. From the convexity of $\mathcal{WO}_i$ it implies that $T \in \mathcal{WO}_i$. We have that $\overrightarrow{AT} = \overrightarrow{AN} + t\overrightarrow{NW}$, $t \in [0, 1]$, and $\overline{AT}^2 = d_{ij}^2 + t^2\overrightarrow{NW}^2 + 2t\overrightarrow{AN}\left(\overrightarrow{AW} - \overrightarrow{AN}\right) = d_{ij}^2 + t^2\overrightarrow{NW}^2 - 2td_{ij}^2 + 2td_{ij}x\cos\delta = \overrightarrow{NW}^2t^2 + 2td_{ij}\left(x\cos\delta - d_{ij}\right) + d_{ij}^2 \equiv f(t)$. Observe the behavior of the quadratic function $f(t)$ in some vicinity of $t = 0$, where $t \geq 0$. Clearly, $f(0) = d_{ij}^2 = \overline{NA}^2$. The first derivative the function $f(t)$ is $\frac{df(t)}{dt} = 2t\overrightarrow{NW}^2 + 2d_{ij}\left(x\cos\delta - d_{ij}\right)$. Since $x\cos\delta < d_{ij}$, the limit value of $f(t)$ at $t = 0_+$ is clearly negative, i.e., $\lim_{t\searrow 0}\frac{df(t)}{dt} = 2d_{ij}\left(x\cos\delta - d_{ij}\right) < 0$, which means that for some sufficiently small $\epsilon > 0$, $f(t)$ decreases in $t \in [0, \epsilon]$ meaning that $f(\epsilon) < f(0) = d_{ij}^2$. Thus, there exists a point $T \in \overline{NW}$ such that $\overline{AT}^2 < d_{ij}^2$. This contradicts the fact that the minimum distance between $\overline{PQ}$ and $\mathcal{WO}_i$ is $\rho\left(\overline{PQ}, \mathcal{WO}_i\right) = d_{ij}$, which completes the proof. ■

REFERENCES

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

[2] S. M. LaValle and J. J. Kuffner, "Rapidly-exploring random trees: Progress and prospects," in *Proceedings Workshop on the Algorithmic Foundations of Robotics*, 2000.

[3] J. Kuffner, J.J. and S. LaValle, "RRT-connect: An efficient approach to single-query path planning," in *Robotics and Automation, 2000. Proceedings. ICRA '00. IEEE International Conference on*, vol. 2, 2000, pp. 995–1001.

[4] S. Karaman and E. Frazzoli, "Sampling-based algorithms for optimal motion planning," *International Journal of Robotics Research*, vol. 30, no. 7, pp. 846–894, June 2011.

[5] L. Jaillet and J. Porta, "Asymptotically-optimal path planning on manifolds," in *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[6] D. Berenson, S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *International Journal of Robotics Research (IJRR)*, vol. 30, no. 1, pp. 1435 – 1460, October 2011.

[7] S. Redon, M. C. Lin, D. Manocha, and Y. J. Kim, "Fast continuous collision detection for articulated models," *Journal of Computing and Information Science in Engineering*, vol. 5, no. 2, pp. 126–137, 2005.

[8] J. Kwon and O. Khatib, "Adaptive collision checking for continuous robot motions within motion constraints," in *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*. IEEE, 2013, pp. 5365–5372.

[9] N. Vahrenkamp, T. Asfour, and R. Dillmann, "Efficient motion planning for humanoid robots using lazy collision checking and enlarged robot models," in *Intelligent Robots and Systems, 2007. IROS 2007. IEEE/RSJ International Conference on*, Oct 2007, pp. 3062–3067.

[10] D. Bertram, J. Kuffner, R. Dillmann, and T. Asfour, "An integrated approach to inverse kinematics and path planning for redundant manipulators," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 15-19 2006, pp. 1874 –1879.

[11] S. Quinlan, "Real-time modification of collision-free paths," Ph.D. dissertation, Stanford University, 1995.

[12] B. Lacevic and P. Rocco, "Safety-oriented path planning for articulated robots," *Robotica*, vol. 31, no. 06, pp. 861–874, 2013.

[13] O. Salzman, K. Solovey, and D. Halperin, "Motion planning for multilink robots by implicit configuration-space tiling," *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 760–767, 2016.

[14] A. Tahirovic and M. Ferizbegovic, "Rapidly-exploring random vines (RRV) for motion planning in configuration spaces with narrow passages," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7055–7062.

[15] A. C. Shkolnik and R. Tedrake, "Sample-based planning with volumes in configuration space," *CoRR*, vol. abs/1109.3145, 2011.

[16] J. Bialkowski, M. Otte, S. Karaman, and E. Frazzoli, "Efficient collision checking in sampling-based motion planning," in *Algorithmic Foundations of Robotics (WAFR), 10th International Workshop on*, 2013, pp. 365–380.

[17] A. Knobloch, N. Vahrenkamp, M. Wächter, and T. Asfour, "Distance-aware dynamically weighted roadmaps for motion planning in unknown environments," *IEEE Robotics and Automation Letters (RA-L)*, pp. 0–0, 2018.

[18] B. Lacevic, D. Osmankovic, and A. Ademovic, "Burs of free C-space: a novel structure for path planning," in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE, 2016, pp. 70–76.

[19] J.-M. Lien and N. M. Amato, "Approximate convex decomposition of polyhedra and its applications," *Computer Aided Geometric Design*, vol. 25, no. 7, pp. 503–522, 2008.

[20] K. Mamou and F. Ghorbel, "A simple and efficient approach for 3d mesh approximate convex decomposition," in *Image Processing (ICIP), 2009 16th IEEE International Conference on*. IEEE, 2009, pp. 3501–3504.

[21] A. Gaschler, R. P. A. Petrick, T. Kröger, A. Knoll, and O. Khatib, "Robot task and motion planning with sets of convex polyhedra," in *Workshop on Task and Motion Planning at Robotics Science and Systems*, Berlin, Germany, June 2013.

[22] F. Schwarzer, M. Saha, and J.-C. Latombe, "Exact collision checking of robot paths," in *Algorithmic foundations of robotics V*. Springer, 2004, pp. 25–41.

[23] ——, "Adaptive dynamic collision checking for single and multiple articulated robots in complex environments," *IEEE Transactions on Robotics*, vol. 21, no. 3, pp. 338–353, 2005.

[24] S. Tarbouriech and W. Suleiman, "On bisection continuous collision checking method: Spherical joints and minimum distance to obstacles," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 7613–7619.

[25] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, May 2012, pp. 3859–3866.

[26] M. Muja and D. G. Lowe, "Fast approximate nearest neighbors with automatic algorithm configuration," in *International Conference on Computer Vision Theory and Application VISSAPP'09)*. INSTICC Press, 2009, pp. 331–340.

[27] B. Lacevic, D. Osmankovic, and A. Ademovic, "Path planning using adaptive burs of free configuration space," in *2017 XXVI International Conference on Information, Communication and Automation Technologies (ICAT)*. IEEE, 2017, pp. 1–6.

[28] I. A. Sucan, M. Moll, and L. E. Kavraki, "The open motion planning library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, 2012.