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

Constraint Manifold Exploration for Efficient Continuous Coverage Estimation

Robert Wilbrandt[1] and Rüdiger Dillmann[1]

***Abstract*— Many automated manufacturing processes rely on industrial robot arms to move process-specific tools along workpiece surfaces. In applications like grinding, sanding, spray painting, or inspection, they need to cover a workpiece fully while keeping their tools perpendicular to its surface. While there are approaches to generate trajectories for these applications, there are no sufficient methods for analyzing the feasibility of full surface coverage. This work proposes a sampling-based approach for continuous coverage estimation that explores reachable surface regions in the configuration space. We define an extended ambient configuration space that allows for the representation of tool position and orientation constraints. A continuation-based approach is used to explore it using two different sampling strategies. A thorough evaluation across different kinematics and environments analyzes their runtime and efficiency. This validates our ability to accurately and efficiently calculate surface coverage for complex surfaces in complicated environments.**

I. INTRODUCTION

Industrial robot arms are a vital part of modern automated manufacturing operations. An extensive range of tasks requires them to move process-specific tools along predefined workpiece surfaces. The chosen trajectory directly influences the process result: Sanding and polishing rely on uniform tool engagement, spray painting requires a consistent coating, and automated inspection should move continuously to ensure efficient cycle times [1]. All of these tasks must not only cover the entire surface but also keep the tool orthogonal to the surface.

There is a significant body of work on planning trajectories for these applications as constrained *Coverage Path Planning* (CPP) problems. In surface finishing, several approaches plan for contact areas between tool and surface [2] and incorporate segmentation for complex parts [3]. Automated dimensional inspection requires smooth trajectories to optimize accuracy [4]. In automated depowdering, covering paths need to address highly complex geometries efficiently [5]. For general coverage planning, local parametrizations of surface geometries were proposed [6], and UV grids for specific types of surfaces can be calculated [7]. Projection methods can apply 2D geometry to complex surfaces [8], and graph solvers can prevent redundant coverage [9]. While these approaches generally try to cover the entire workpiece geometry, they cannot determine the feasibility of covering a given surface for a specific scenario.

The general problem of trajectory planning with constraints has received significant attention in the literature [10]. In addition to being able to plan motions along such surfaces, they can incorporate robot dynamics [11] or handle changing constraints across the environment [12]. They are, however, only able to plan single motions and cannot give any indication of coverage feasibility.

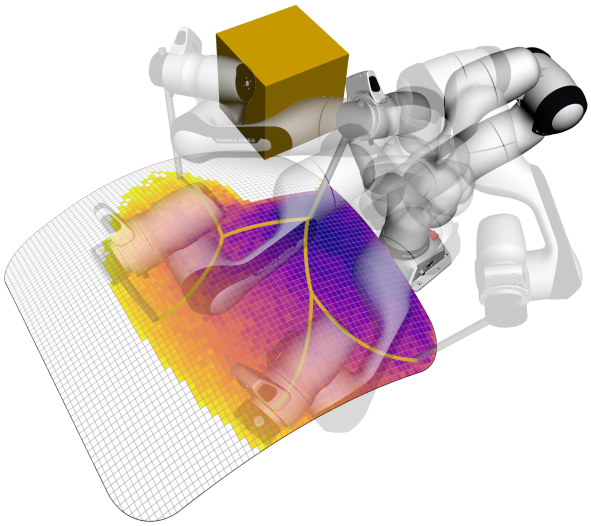


Fig. 1. Continuous coverage of a surface by a robot using the proposed approach. Starting from an initial configuration, continuous paths across the surface are explored. Reachable surface cells are colored based on their exploration order and some configurations near the border are overlaid.

Closely related to coverage feasibility, reachability analysis is a well-known robotic problem. Reachability maps and inverse reachability maps can be used for placement optimization [13], and advanced maps can analyse nullspace motion in redundant manipulators [14]. The generation of reachability maps can be optimized using $SE(3)$ convolutions [15], and dimensionality reductions can be used to reduce computation and storage requirements for common robot kinematics significantly. All of these approaches can help determine if and how a pose on a surface is reachable. They can, however, not decide if this is possible in a continuous motion on the surface without breaking the tool constraints.

There is a prior approach for estimating continuous surface coverage [16] that was subsequently extended to consider the robot jacobian for improved exploration performance [17]. While their approach is highly efficient, it relies on discretizations of the surface and the configuration space and thus cannot express constraints exactly. We also argue in Section IV that their approach can underestimate coverage in the presence of obstacles.

This work proposes a novel approach for robot continu-

[1]FZI Research Center for Information Technology, Haid-und-Neu-Straße 10–14, 76131 Karlsruhe, Germany {wilbrandt,dillmann}@fzi.de

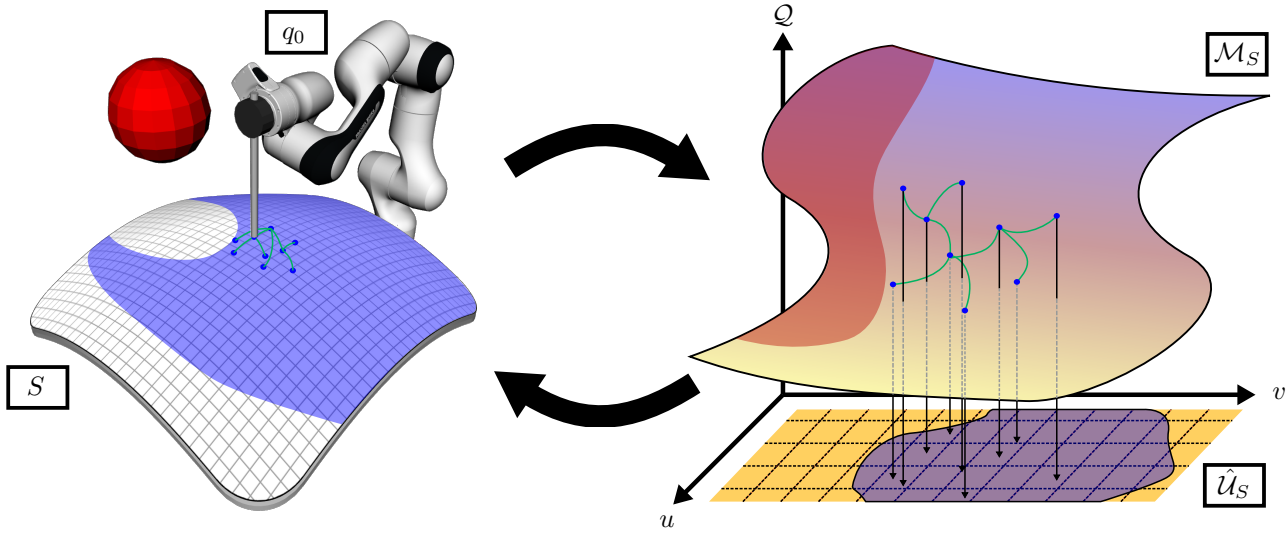


Fig. 2. Continuous coverage estimation by configuration space manifold exploration. Continuous motions of the robot on the left along the surface S form a smooth manifold $\mathcal{M}_S$ on the right. Starting from an initial configuration q_0, we explore the manifold using the configuration-space motions shown in green and project them to surface coordinates. Continually doing so increases the estimated coverable region $\hat{\mathcal{U}}_S$ until the blue region on the left is covered by samples.

ous coverage estimation based on randomized sampling of implicit constraint manifolds that can be seen in Figure I. It can work on complex free-form surfaces and adhere exactly to tool constraints to obtain accurate coverage results.

The remainder of this paper is organized as follows. Section II formally defines the continuous coverage feasibility problem. In Section III, we present an approach based on the exploration of an implicit constraint manifold and introduce two exploration strategies. We compare them across a range of different kinematics and scenarios and analyze their performance in Section IV. Finally, we discuss straightforward extensions to the approach, practical limitations, and avenues for future research in Section V.

II. PROBLEM FORMULATION

Consider a robot system with an n-dimensional configuration space $\mathcal{Q}$ with $\mathcal{Q}_{\text{free}} \subseteq \mathcal{Q}$ being the set of non-colliding configurations. For a given configuration $q \in \mathcal{Q}$, we refer to $f_{\text{pos}}(q) \in \mathbb{R}^3$ as the tool position and $f_{\text{rot}}(q) \in \mathbb{R}^{3\times3}$ as the rotation matrix representation of the robot tool orientation. By convention and without loss of generality, the orientation of the tool axis is assumed to be given by its z-axis $f_{\text{rot}}(q) \cdot \begin{pmatrix}0 & 0 & 1\end{pmatrix}^T$.

In addition to the robot, we need to consider the surface S to be evaluated. Without constraints on a specific representation, we assume the surface to be a C^2-smooth function $S : \mathbb{R}^2 \to \mathbb{R}^3$ and only consider a subset of the surface domain $\mathcal{U}_S \subseteq [u_{\min}, u_{\max}] \times [v_{\min}, v_{\max}]$. We write $n_S(u,v) = \frac{\delta S}{\delta u} \times \frac{\delta S}{\delta v}$ to refer to the surface normal.

When estimating the coverage of the surface, only configurations with tool positions on S and tool axes orthogonal to S should be considered, leading to the definition

Definition 2.1 (Surface-Constrained Configuration): We consider a configuration $q \in \mathcal{Q}$ *surface-constrained* with regards to a surface S if there is a surface coordinate $(u,v) \in \mathcal{U}_S$ with $f_{\text{pos}}(q) = S(u,v)$ and $(f_{\text{rot}}(q)\begin{pmatrix}0 & 0 & 1\end{pmatrix}^T) \cdot \hat{n}_S(u,v) = 1$.

We write $\mathcal{Q}_S \subseteq \mathcal{Q}$ to refer to the set of all surface-constrained configurations and $\mathcal{Q}_S^\star = \mathcal{Q}_S \cap \mathcal{Q}_{\text{free}}$ for the subset that is not in collision.

For robot trajectories $\sigma : [0,1] \to \mathcal{Q}$ where each $\sigma(t)$ is surface-constrained with regards to S, there is a surface path $\sigma_S : [0,1] \to \mathcal{U}$ such that $\sigma(t)$ satisfies all motion constraints with regards to $\sigma_S(t)$ for all $t \in [0,1]$. We can use this to define the problem

Definition 2.2 (Continuous Coverage Estimation): Given an initial configuration $q_{\text{start}} \in \mathcal{Q}_S^\star$, find the set $\hat{\mathcal{U}}_S \subseteq \mathcal{U}_S$ of surface coordinates $(u_{\text{cc}}, v_{\text{cc}})$ for which a trajectory $\sigma^\star : [0,1] \to \mathcal{Q}_S^\star$ exists with $\sigma_S^\star(1) = (u_{\text{cc}}, v_{\text{cc}})$.

III. APPROACH

We propose a sampling-based approach for estimating continuous coverage, as shown in Figure I. For this, we define an extended configuration space $\mathcal{Q}^+$ that can directly express the constraints on tool position and orientation. The resulting implicit manifold directly maps to $\mathcal{Q}_S$, enabling the exploration of continuously reachable surface coordinates through continuous configuration-space motions. We present two methods for its exploration. Both of them perform probabilistic sampling and run continuously, but at any point in time, we can stop and evaluate the explored samples and estimate the continuously covered subset of $\hat{\mathcal{U}}_S$.

A. Constrained Configuration Space

In order to construct the space of all surface-constrained configurations, we first create an ambient space consisting of robot configurations and surface coordinates. We can

Algorithm 1 RRT Exploration of Constraint Manifold

Input: Initial Robot Configuration $q_0 \in \mathcal{Q}_{\text{free}}$

Find closest point $S(u_0, v_0)$ to $f_{\text{pos}}(q_0)$
Project (q_0, u_0, v_0) to obtain $q_S \in \mathcal{M}$
Initialize tree T with root q_S
loop
  $q \leftarrow \text{SAMPLEUNIFORM}(\mathcal{M})$
  $q_N \leftarrow \text{NEARESTNEIGHBOR}(q, T)$
  if $\text{dist}(q, q_N) > d_{\max}$ **then**
    $q \leftarrow \text{GEODESICINTERPOLATE}(q_N, q, \frac{d_{\max}}{\text{dist}(q, q_N)})$
  end if
  if $\text{CHECKTRANSITION}(q_N, q, \delta q_{\text{check}})$ **then**
    $T \rightarrow \text{ADDCHILD}(q, q_N)$
  end if
end loop

Algorithm 2 Biased Manifold Exploration

Input: Initial Robot Configuration $q_0 \in \mathcal{Q}_{\text{free}}$

Find closest point $S(u_0, v_0)$ to $f_{\text{pos}}(q_0)$
Project (q_0, u_0, v_0) to obtain $q_S \in \mathcal{M}$
Initialize empty grid G
$G \rightarrow \text{ADD}(q_S, (u_0, v_0))$
loop
  $q \leftarrow \text{SELECT}(G)$
  $q' \leftarrow \text{SAMPLEGAUSSIAN}(\mu = q, \sigma = \sigma_{\text{sample}})$
  if $\text{CHECKTRANSITION}(q, q', \delta q_{\text{check}})$ **then**
    $G \rightarrow \text{ADD}(q', [q]_{uv})$
  end if
end loop

formalize this as

$$\mathcal{Q}^+ = \{\, (q, u, v) \mid q \in \mathcal{Q}, (u, v) \in \mathcal{U} \,\} \tag{1}$$

We want to constrain this space to only configurations (q, u, v) where the robot configuration q adheres to tool position- and orientation constraints at surface position (u, v). For both constraints, we define constraint functions that are zero if their constraint is fulfilled. For the position constraint, we achieve this using

$$C_{\text{pos}} = f_{\text{pos}}(q) - S(u, v) \tag{2}$$

For the tool orientation constraint, we consider the surface normal in the coordinate frame of the robot tool. For constraint-satisfying configurations, this should be colinear with the z-axis. We can thus define

$$C_{\text{rot}} = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix} f_{\text{rot}}^{-1}(q) \cdot \mathbf{n}_S(u, v) \tag{3}$$

While this also allows configurations where the robot tool opposes the surface normal, this is not a problem in this application, as a transition to this *flipped* state is impossible.

The combined constraint function $C = \begin{bmatrix} C_{\text{pos}} & C_{\text{rot}} \end{bmatrix}^T$ implicitly defines the constrained configuration space

$$\mathcal{M} = \{q^+ \in \mathcal{Q}^+ \mid C(q) = \mathbf{0}\} \tag{4}$$

By construction, this space is a smooth manifold in $\mathcal{Q}^+$ and directly maps to $\mathcal{Q}_S$. While we generally cannot parametrize it globally, we can explore C's nullspace using its jacobian to explore it gradually. With $\mathcal{Q}^+$ being of dimension $(n + 2)$, $\mathcal{M}$ should be of dimension $(n - 3)$.

B. Constraint Manifold Exploration

For any given configuration in M, the kernel of the jacobian of C can be used to construct a local tangent space that acts as a linear approximation in a small neighborhood. New valid configurations can be found by sampling in this hyperplane and projecting back to M. We utilize the IMACS [18] framework for this, which interleaves the sampling of new configurations with the creation of new hyperplanes. In this context, these tangent spaces are referred to as *charts* and the set of all charts as *atlas*. IMACS allows us to sample new configurations from a given atlas, project configurations to $\mathcal{M}$, and interpolate between states along geodesics. We refer to Kingston et al. [18] for a detailed description.

Building on IMACS, we propose two algorithms for manifold exploration. First, we create an RRT that utilizes uniform sampling over all previously covered charts. This acts as a solid baseline and can provide asymptotic coverage guarantees, but can be inefficient without any sample biasing. To address this, we introduce a second algorithm based on the KPIECE [19] planner that guides exploration towards unknown regions of the S.

1) RRT-Based Exploration: Our first method utilizes uniform sampling over the covered atlas to gradually grow the covered region of M and is shown in Algorithm 1. For its initialization, it first finds the closest surface coordinate (u_0, v_0) to $f_{\text{pos}}(q_0)$ and then projects (q_0, u_0, v_0) to $\mathcal{M}$. The resulting configuration $q_S \in \mathcal{Q}^+$ is used as the root of the constructed tree T.

The iterative exploration follows the standard RRT procedure. Using IMACS, we first sample a random configuration q using the already covered charts, rejecting states outside the joint limits or in collision. We then find its closest know neighbor q_N in T and use the geodesic between q_N and q to expand T in the direction of q. To ensure continuous motions, we check the transition between them at a regular step of δq_{check} and make sure that interpolation is possible and does not lead to collisions.

Following the argument by Kingston et al. [18], the proof by Jaillet and Porta [20] for the eventual coverage of $\mathcal{M}$ by the atlas created by IMACS holds. We therefore argue that for any subset $\mathcal{U}' \subseteq \mathcal{U}$ of non-zero measure that is continuously reachable from q_0, we will eventually sample a configuration (q, u, v) with $(u, v) \in \mathcal{U}'$. Therefore, we are guaranteed to eventually estimate the continuously coverable region exactly using this approach.

2) Biased Sampling: While uniform RRT-based exploration has nice theoretical properties and will eventually result in an accurate estimate, it can perform poorly in practice. This is especially true in cluttered environments with narrow passages, where sampling-based planners traditionally also

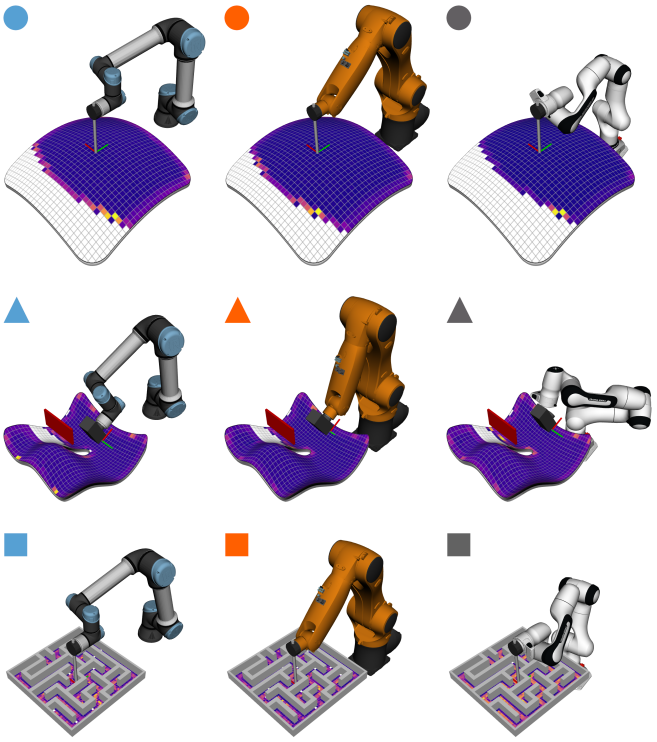


Fig. 3. The matrix of robots and environments used throughout the evaluation. We use two six-axis and one seven-axis robots to cover different relevant kinematics, and three surfaces with different curvatures and obstacles to achieve representative results. The icons next to each robot are used throughout the evaluation to reference specific setups.

struggle [21].

To combat these limitations, we propose a biased sampling exploration scheme based on a 2D projection and the KPIECE [19] sample biasing scheme that can be seen in Algorithm 2. It maintains a 2D grid to bias expansion towards unexplored regions, preventing excessive sampling in easily reachable areas. We index the grid based on each configuration $q \in \mathcal{M}$'s surface coordinates $[q]_{uv}$ and store all newly sampled configurations in their respective cell. The grid keeps track of *interior* and *exterior* cells depending on samples in neighboring cells, selecting an *exterior* cell for expansion with a strong bias. Both types of cells are ranked according to the *Importance* of a cell z as defined by

$$\frac{\log(\mathcal{I}) \cdot score}{\mathcal{S} \cdot (1 + |Neighbors(z)|) \cdot Coverage(z)} \tag{5}$$

with $\mathcal{I}$ being the iteration at which z was first visited, $score$ tracking the progress previously achieved when expanding from it and $\mathcal{S}$ being the number of times it was previously expanded. We refer to Sucan et al. [22] for a detailed description of cell *Importance* and the cell selection process.

Just as the RRT-based exploration approach, the initial configuration q_0 is first projected to obtain $q_S \in \mathcal{M}$. The grid is then initialized and q_S is stored in G at $[q_S]_{uv}$. Starting in line 6, sampling is now performed by selecting a state q in G and sampling around it using a Gaussian distribution to obtain q'. The geodesic between q and q' is then checked at regular intervals δq_{check} for joint limit validity and collisions, and added to G if the transition turned out to be valid.

C. Coverage Estimation

After any number of samples, we can analyze all configurations sampled so far to obtain an approximation for the continuously coverable region of S. For evaluation, we discretize $\mathcal{U}$ into an equally spaced $n_{\text{grid}} \times n_{\text{grid}}$ grid and record the number $v_{i,j}$ of configurations that visited each cell. We report the set of all cells with $v_{i,j} \geq 0$ as continuously reachable. For the biased exploration approach described in Section III-B.2, we use the same grid size for cell selection as for coverage estimation.

While n_{grid} defines the maximum achievable result resolution, it does not affect the accuracy of constraint satisfaction in the prior exploration step. In particular, a coarse value of this parameter can not lead to incorrectly identified connections between otherwise disconnected surface regions.

IV. RESULTS AND EXPERIMENTS

The proposed approach was implemented using the constraint planning capabilities [18] of the *Open Motion Planning Library* [22]. We represent complex surfaces using the *OpenCASCADE* CAD geometry kernel and utilize the *MoveIt* framework [23] for collision detection. All benchmarks are performed on a desktop PC with an AMD Ryzen 9 9950X 16-core CPU running at a base frequency of 4.3 GHz with 64 GB of DDR5 memory. As the evaluation primarily stresses a single CPU Core, we run up to five instances in parallel.

We start the evaluation by looking at the impact of the parameters $d_{\max}$ and σ_{sample} on runtime and coverage. Based on this, we select specific parameter values and analyze the sampling behavior and its impact on sample efficiency. Finally, we analyze the required runtime to achieve accurate estimates.

A. Sampling parameters

To investigate the effect of exploration parameters on the runtime and accuracy of the proposed approach, we use a test matrix composed of different robots and environments shown in Figure 3. For practical relevance, we use two robots with six degrees of freedom (Universal Robots UR5e, Kuka KR10 R900 sixx) and one with seven degrees of freedom (Franka Research 3). For each of them, we evaluate three environments of different complexity. In the simplest case, we have a symmetric surface with low curvature and without obstacles, focusing on the pure kinematic reachability. In an advanced scenario, we have much higher surface curvatures and an additional obstacle that blocks transitions along one side of the surface. Finally, an 8×8 maze is used to study the effect of narrow passages.

All experiments in this chapter are run with $\delta_{\text{check}} = 0.01$. Unless otherwise noted, we use $n_{\text{grid}} = 32$ for scenarios 1 and 2, and $n_{\text{grid}} = 48$ for scenario 3. As a baseline for the experiments, we performed exhaustive sampling of

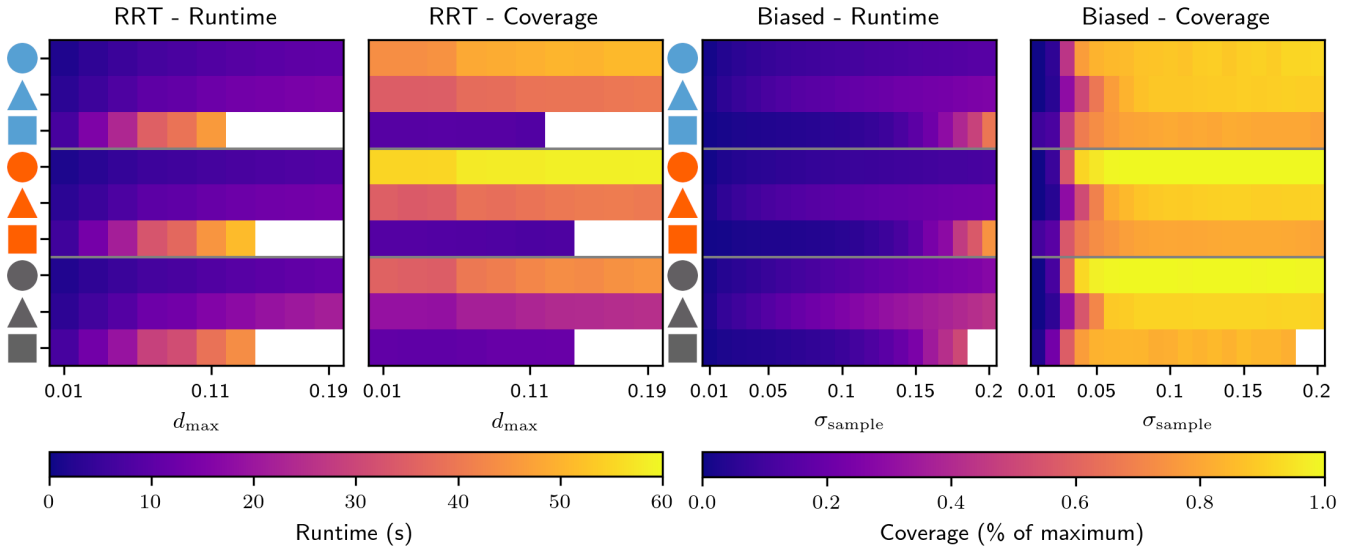
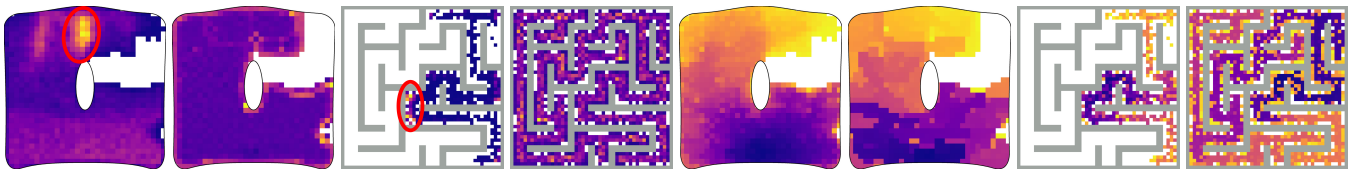


Fig. 4. The results of the parameter evaluation. For both exploration approaches, the mean runtime and the mean coverage in percent of reachable cells across 25 experiments of 25 000 samples are shown. The rows are organized in groups by scenario.



(a) RRT (b) Biased (c) RRT (d) Biased (e) RRT (f) Biased (g) RRT (h) Biased

Fig. 5. Sampling behavior of the Kuka robot in the curved and maze environments after 100 000 samples. Figures (a) – (d) visualize the number of samples per cell, while (e) – (h) show the order of exploration. Spots of high sampling density were manually annotated.

10 million configurations to obtain a baseline of reachable cells for each robot and scenario.

We perform experiments with both proposed exploration approaches on each combination of robot and scenario. For the RRT-based exploration approach, we use different values of $d_{\max}$ within $[0.01, 0.19]$ in 10 steps of 0.02. For the biased exploration approach, we test σ_{sample} values in $[0.01, 0.20]$ in steps of 0.01 and use a constant exterior cell bias of 75% as in the initial KPIECE publication. Each experiment was run 25 times with a timeout of $60\,\text{s}$ per run. The results of these experiments can be seen in Figure 4.

While the RRT-based and biased exploration perform comparably in the simple scenario with six degrees of freedom, biased exploration is significantly more performant and sample efficient in any more complex scenario. In particular, the maze scenario is barely solvable using RRT exploration and will not be considered in further analysis. As expected, a larger step or sample size generally leads to faster coverage. With the increase, the probability of being unable to project a sample from a chart to $\mathcal{M}$ also increases. Together with our approach only registering the samples themselves and not the geodesics towards them, this increases the required sample time. Across different robots, the behavior seems pretty similar, with the Franka robot achieving less coverage due to its higher-dimensional search space. Both runtime and coverage across different scenarios differ quite a bit, which is to be expected due to the larger required joint motion in the curved scenario and the highly restricted environment in the maze scenario. For this approach to be practical for real-world analysis of robot continuous reachability, we do, however, need to be able to use one set of parameters for any given surface and robot. Based on the tradeoff between coverage efficiency and required runtime, we manually choose $d_{\max} = 0.07$ for RRT-based exploration and $\sigma_{\text{sample}} = 0.04$ for biased exploration and will use these values for the remainder of the evaluation.

B. Sampling Behavior

In order to analyse the discrepancy in sampling performance between the RRT-based exploration and biased exploration, we ran exploration for $100\,000$ samples for the Kuka robot in the curved and maze environments and observed the number of samples per cell on the surface and the order in which exploration occurred. The results of this can be seen in Figure 5.

First of all, we observe the density of samples per cell. The biased approach leads to a generally uniform distribution without particular high spots. Only the borders have a higher density, where the biased sampling tried to extend further and could not find valid new neighbor cells. In contrast, one can clearly see spots of very high density for the RRT-based

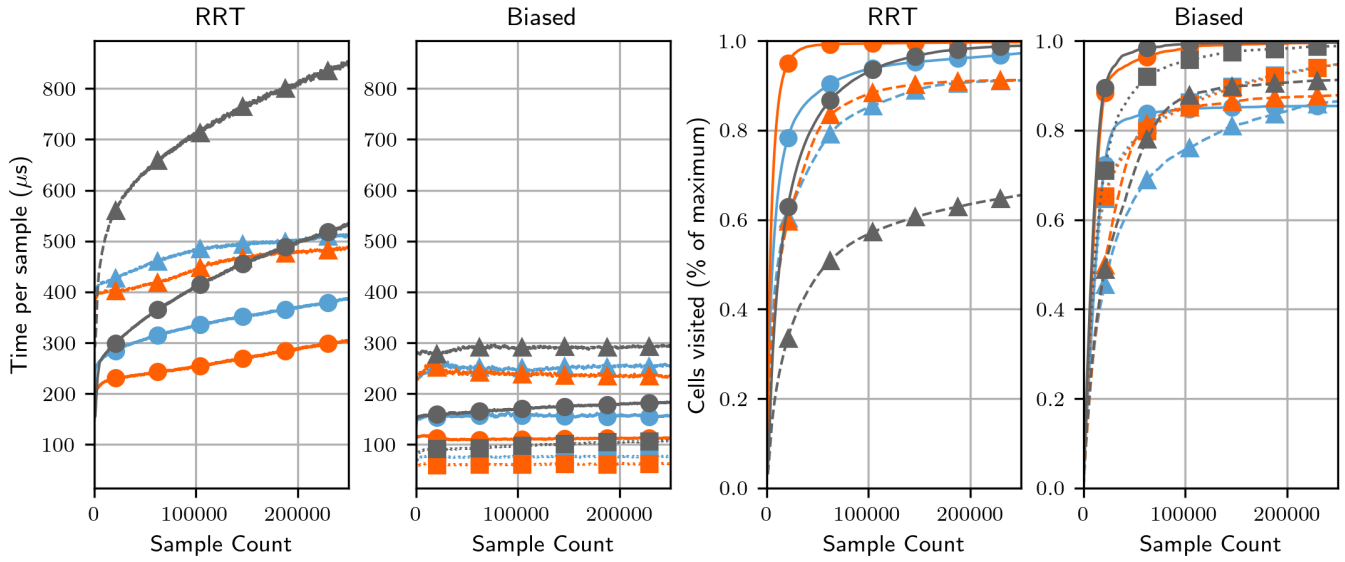


Fig. 6. Evaluation of mean sample time (left) and mean coverage (right) during exploration with 250 000 samples. The sample times were smoothed using a bidirectional exponential moving average (EMA, $\alpha = 0.01$).

exploration. Expansion of the RRT happens mostly uniformly in $\mathcal{M}$, which does not translate to a uniform distribution on S. In the curved scenario that is especially visible, as the high-density region contains configurations close to a robot singularity, where steps in axes 4 and 6 will lead to minimal motion on the surface.

The expansion behavior is also visible in the expansion progression. The RRT-based exploration gradually expands in all directions, without any discernible jumps in the resulting figure. The biased sampling instead shows regions that were expanded rapidly, showing large jumps to adjacent regions that were only discovered later. This is a clear result of the $\frac{\log(\mathcal{I})}{S}$ scoring in (5), which leads to newly covered cells being preferred for the next sampling steps. In the maze environment in particular, one can see sections being explored at once before switching to a completely different part of the surface.

It is important to note that complete coverage of $\mathcal{M}$ is computationally expensive and is often not required for an accurate estimate. In the presence of obstacles and complex surfaces, it is, however, not generally possible to get accurate estimates without any exploration of the inherent kinematic redundancy of the problem. We see this as the major contribution of the proposed concept and the main contribution compared to greedy search approaches in prior work [16], [17], which only perform a 2D grid search.

C. Performance

As a final evaluation step, we analyse the performance of the proposed approach. For this, we first look at sample time and coverage over the number of samples, before evaluating the average times to achieve coverage percentiles.

We run the exploration on all robots and scenarios for 100 iterations at 250 000 samples each. The resulting mean time per sample and mean coverage at each step can be seen in Figure 6.

The time taken to generate a sample for RRT-based exploration increases with the number of total samples, while the time taken for biased exploration seems mostly constant. We attribute this mainly to the Nearest-Neighbor structure needed for RRT expansion, which is not required in the KPIECE weighting scheme. This matches the more substantial time increase for the 7-DOF Franka arm in the RRT case, where an additional dimension needs to be tracked. Comparing scenarios, the curved surface takes significantly more effort than the slightly rounded surface. Perhaps counter-intuitively, the maze scenario (only considered for the biased case as discussed in Section IV-A) takes an even smaller time per sample than the simple curved scenario. The simple structure of the surface allows for very fast sampling, and due to the obstacle structure, the probability of discarding samples due to collisions stays approximately constant throughout the experiment.

Regarding coverage per sample, the RRT-based exploration seems to arrive at a high level of coverage with fewer samples than the biased approach. This is, however, notably different for the 6-DOF Franka curved scenario, where a lot of nullspace motion is required to reach further parts of the surface. Perhaps surprisingly, the simple scenario (especially for the UR robot) performs worse for biased sampling than RRT-based exploration. We suspect this to be caused by the border of coverage on the surface, where the inherent IMACS biasing towards open charts leads to exhaustive coverage of all boundary cells. At the same time, the biased exploration at some point no longer finds new cells near the border and thus searches in different parts.

While the prior steps have looked at estimation speed, they focused on per-sample performance and efficiency. As a final step, we analyze the coverage estimate over the runtime of the approach. For this, we run 100 iterations using the Kuka

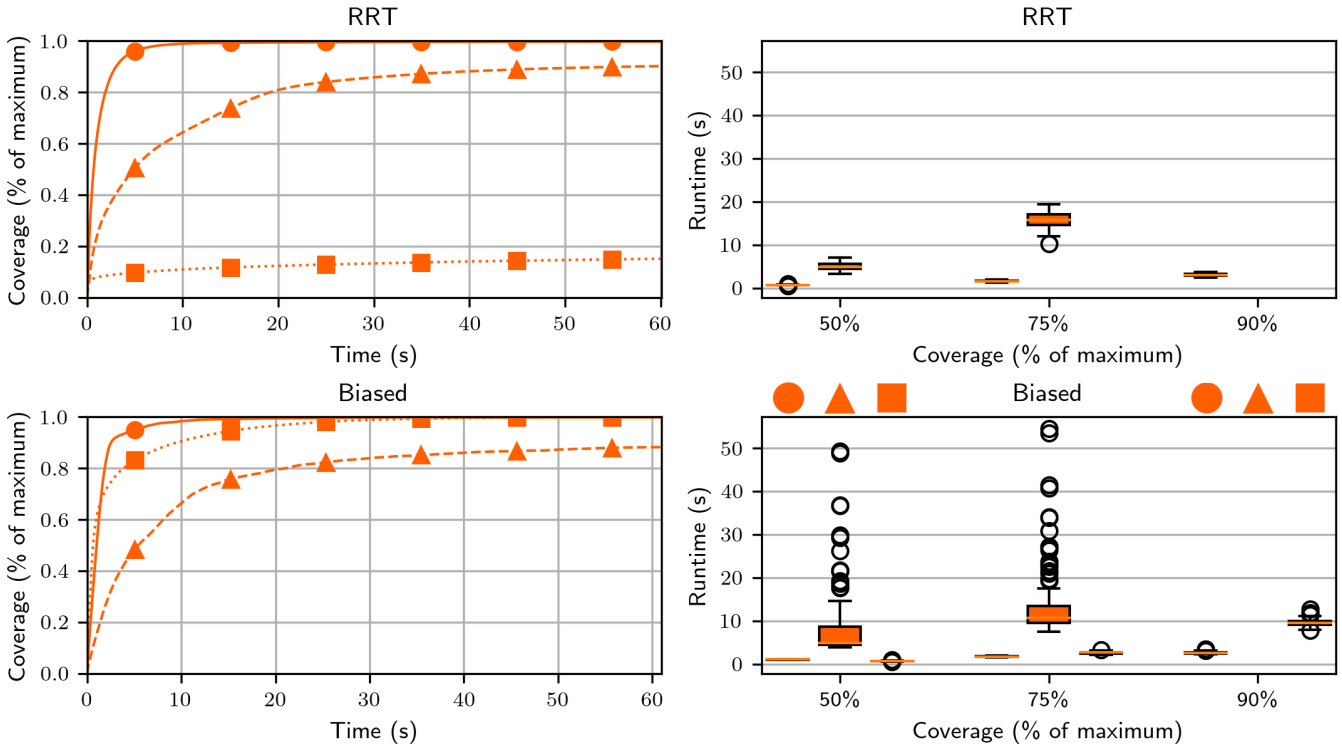


Fig. 7. Coverage over time and time to achieve a specific degree of coverage. All experiments were done 100 times, with the left plots showing the mean coverage at every point in time. Boxes for scenarios in which coverage to a certain degree was not consistently possible were omitted in the right plot.

robot across all scenarios, the results of which can be seen in Figure 7.

Both approaches perform significantly better on the simple scenario than the curved one, reaching $90\,\%$ coverage on average within the first three seconds. The curved scenario requires more time, achieving $80\,\%$ coverage only within 19 and 21 seconds. Interestingly, the RRT-based exploration performs better in this case, but neither approach can consistently surpass $90\,\%$ coverage in the allotted time. For the lower curvature scenarios, biased exploration is faster, but has a much larger spread with significant execution time outliers, while RRT-based exploration performs more predictably.

V. CONCLUSIONS AND FUTURE WORK

We have presented a novel approach for continuous coverage estimation using an implicit manifold constraint. By transferring robot motions to an extended configuration space, we can capture the structure of valid motions and iteratively explore the complete robot action space. Two sampling approaches are proposed and compared for its exploration and evaluated across a range of robots and scenarios.

We show that this general scheme can provide accurate estimates of the continuously reachable surface regions, which can be used as a building block for higher-level cell optimization or as a starting point for region-based tool path generation. Its ability to work directly on complex CAD surfaces makes it applicable in real-world industrial contexts. While its functionality is currently limited by its requirements on surface smoothness, straightforward extensions could project samples across discontinuities and expand the possible scope of application.

The current approach provides a purely kinematic analysis. Future work should expand on this to incorporate motion smoothness and robot dynamics. Furthermore, we plan to investigate further uses of manifold exploration for direct coverage path planning and enhanced robot performance metrics.

REFERENCES

[1] E. Glorieux, P. Franciosa, and D. Ceglarek, "Coverage path planning with targetted viewpoint sampling for robotic free-form surface inspection," *Robotics and Computer-Integrated Manufacturing*, vol. 61, p. 101843, 2020.

[2] Y. Wen, D. J. Jaeger, and P. R. Pagilla, "Uniform Coverage Tool Path Generation for Robotic Surface Finishing of Curved Surfaces," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4931–4938, 2022.

[3] S. Schneyer, A. Sachtler, T. Eiband, and K. Nottensteiner, "Segmentation and Coverage Planning of Freeform Geometries for Robotic Surface Finishing," *IEEE Robotics and Automation Letters*, vol. 8, no. 8, pp. 5267–5274, 2023.

[4] Z. Li, K. Tang, P. Hu, and L. Huang, "Five-Axis Trochoidal Sweep Scanning Path Planning for Free-Form Surface Inspection," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 2, pp. 1139–1155, 2023.

[5] V.-T. Do and Q.-C. Pham, "Geometry-Aware Coverage Path Planning for Depowdering on Complex 3D Surfaces," *IEEE Robotics and Automation Letters*, vol. 8, no. 9, pp. 5552–5559, 2023.

[6] Y.-Y. Lin, C.-C. Ni, N. Lei, X. David Gu, and J. Gao, "Robot Coverage Path planning for general surfaces using quadratic differentials," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, 2017, pp. 5005–5011.

[7] S. McGovern and J. Xiao, "UV Grid Generation on 3D Freeform Surfaces for Constrained Robotic Coverage Path Planning," in *2022 IEEE 18th International Conference on Automation Science and Engineering (CASE)*, 2022, pp. 1503–1509.

[8] B. H. Jafari and N. Gans, "Surface Parameterization and Trajectory Generation on Regular Surfaces With Application in Robot-Guided Deposition Printing," *IEEE Robotics and Automation Letters*, vol. 5, no. 4, pp. 6113–6120, 2020.

[9] T. Yang, J. Valls Miro, Y. Wang, and R. Xiong, "An Improved Maximal Continuity Graph Solver for Non-Redundant Manipulator Non-Revisiting Coverage," *IEEE Transactions on Automation Science and Engineering*, vol. 22, pp. 3822–3834, 2025.

[10] 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. Volume 1, 2018, pp. 159–185, 2018.

[11] R. Bordalba, L. Ros, and J. M. Porta, "A Randomized Kinodynamic Planner for Closed-Chain Robotic Systems," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 99–115, Feb. 2021.

[12] Z. Kingston, A. M. Wells, M. Moll, and L. E. Kavraki, "Informing Multi-Modal Planning with Synergistic Discrete Leads," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, May 2020, pp. 3199–3205.

[13] A. Makhal and A. K. Goins, "Reuleaux: Robot Base Placement by Reachability Analysis," in *2018 Second IEEE International Conference on Robotic Computing (IRC)*, 2018, pp. 137–142.

[14] H. Yao, R. Laha, L. F. C. Figueredo, and S. Haddadin, "Enhanced Dexterity Maps (EDM): A New Map for Manipulator Capability Analysis," *IEEE Robotics and Automation Letters*, vol. 9, no. 2, pp. 1628–1635, 2024.

[15] Y. Han, J. Pan, M. Xia, L. Zeng, and Y.-J. Liu, "Efficient SE(3) Reachability Map Generation via Interplanar Integration of Intra-planar Convolutions," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 1854–1860.

[16] S. McGovern and J. Xiao, "Efficient Feasibility Checking on Continuous Coverage Motion for Constrained Manipulation," in *2021 IEEE 17th International Conference on Automation Science and Engineering (CASE)*, Aug. 2021, pp. 189–195.

[17] ——, "A General Approach for Constrained Robotic Coverage Path Planning on 3D Freeform Surfaces," *IEEE Transactions on Automation Science and Engineering*, vol. 21, no. 4, pp. 5546–5557, 2024.

[18] Z. Kingston, M. Moll, and L. E. Kavraki, "Exploring implicit spaces for constrained sampling-based planning," *The International Journal of Robotics Research*, vol. 38, no. 10-11, pp. 1151–1178, 2019.

[19] I. A. Sucan and L. E. Kavraki, "A Sampling-Based Tree Planner for Systems With Complex Dynamics," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 116–131, 2012.

[20] L. Jaillet and J. M. Porta, "Path Planning Under Kinematic Constraints by Rapidly Exploring Manifolds," *IEEE Transactions on Robotics*, vol. 29, no. 1, pp. 105–117, Jan. 2013.

[21] A. Orthey, C. Chamzas, and L. E. Kavraki, "Sampling-Based Motion Planning: A Comparative Review," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 7, May 2024.

[22] I. A. Şucan, M. Moll, and L. E. Kavraki, "The Open Motion Planning Library," *IEEE Robotics & Automation Magazine*, vol. 19, no. 4, pp. 72–82, Dec. 2012.

[23] D. Coleman, I. A. Șucan, S. Chitta, and N. Correll, "Reducing the Barrier to Entry of Complex Robotic Software: a MoveIt! Case Study," *Journal of Software Engineering for Robotics*, vol. 5, no. 1, pp. 3–16, May 2014.