Kinematic Transfer Learning of Sampling Distributions for Manipulator Motion Planning

Peter Lehner; Maximo A. Roa; Alin Albu-Schaeffer

```
@ARTICLE{lehner2022,
 author = {Peter Lehner and Maximo A. Roa and Alin Albu-Schaeffer},
 title = {Kinematic Transfer Learning of Sampling Distributions for Manipulator Motion Planning},
 booktitle={Proc. 2022 IEEE Int. Conf. Robotics and Automation ({ICRA})},
 year={2022},
 pages={7211--7217},
}
```

Kinematic Transfer Learning of Sampling Distributions for Manipulator Motion Planning

Peter Lehner   Máximo A. Roa   Alin Albu-Schäffer[*†]

Abstract

Recent research has shown that guiding sampling-based planners with sampling distributions, learned from previous experiences via density estimation, can significantly decrease computation times for motion planning. We propose an algorithm that can estimate the density from the experiences of a robot with different kinematic structure, on the same task. The method allows to generalize collected data from one source manipulator to similarly designed target manipulators, significantly reducing the computation time for new queries for the target manipulator. We evaluate the algorithm in two experiments, including a constrained manipulation task with five different collaborative robots, and show that transferring information can significantly decrease planning time.

1 INTRODUCTION

Collaborative robots (cobots) have evolved in recent years from research prototypes to a wide range of commercial products. Manufacturers are changing their workflows to incorporate these cobots in their production lines. Based on perception modules, these robots can detect and manipulate objects automatically or in cooperation with operators. To autonomously perform their task, these manipulators require motion planning algorithms that efficiently generate motions within the demanding cycle times in production lines.

Gathering and exploiting data for efficiency in production has become even more relevant. Recent research on motion planning has focused on gaining efficiency by learning from previous instances of a similar task [1, 2, 3], including our own contributions [4, 5]. These approaches explicitly or implicitly assume that the kinematic of the manipulator does not change. In a market where many different cobots are available, this presents a problem: If the manufacturer chooses a particular robot for an application, is the collected data only useful for generating motions for this kind of robot?

In this work we present an algorithm for *transferring* collected data from a source robot to a target robot, increasing the efficiency of motion planning for the new robot on the same task. This technology allows to share experiences between different robots designed for similar applications, a key step in realizing a robot independent database - increasing the flexibility of operators when choosing cobots for their production.

The algorithm transfers previous solutions of the source robot into the configuration space (C-space) of the target robot, and estimates a sampling distribution that guides a sampling-based planner to exploit the relevant C-space of the task. We present an algorithm for transferring the solutions, learning the sampling distribution and biasing a sampling-based planner to solve new problem instances.

We evaluate the algorithm in two experiments, including an experiment that transfers experience for a constrained manipulation task between five cobots, as shown in Figure 1. The experiments show that such experience transfer leads to significant reduction in computation time.

[*]All authors are with the Institute of Robotics and Mechatronics, German Aerospace Center (DLR) at Oberpfaffenhofen, Germany. Contact: peter.lehner@dlr.de

[†]This work was partially funded by the DLR internal project "Factory of the Future Extended"; by the Helmholtz Association, strategic future topic ARCHES ZT-0033, and project COVIPA KA1-Co-02; and the Bavarian Ministry of Economic Affairs, Regional Development and Energy, project OPERA DIK-2107-0005//DIK0374/02

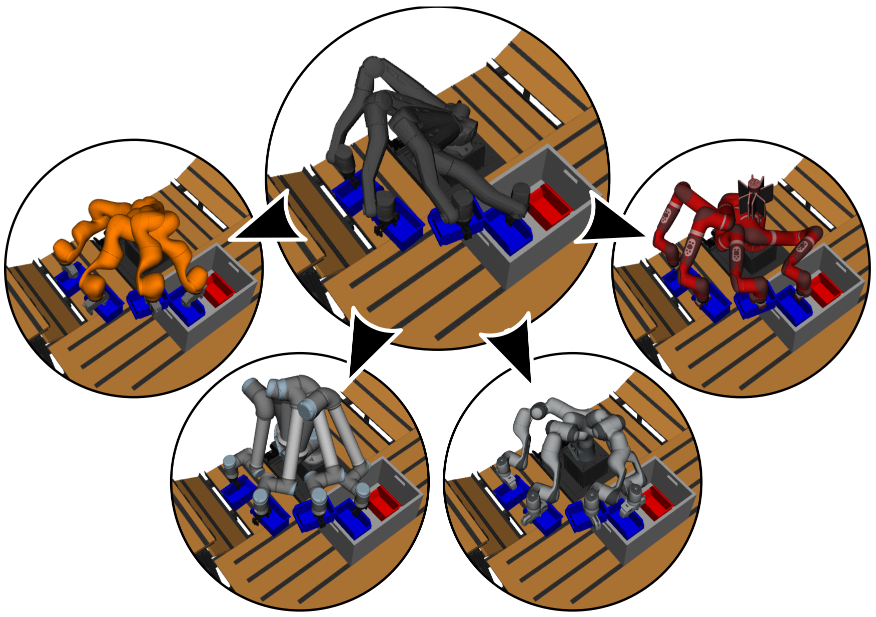

Figure 1: The core idea of kinematic transfer learning: Transfer information from known motions for one type of manipulator into the state space of other manipulators - accelerating the search for motions of the target manipulators.

2 RELATED WORK

Classical sampling-based planning approaches include the Probabilistic Roadmap Method (PRM) [6], which constructs a graph of feasible (collision-free) robot configurations in the C-space for multi-query planning; and the Rapidly-Exploring Random Tree (RRT) [7], which expands a search tree toward unknown regions of the C-space while trying to reach the goal configuration. A derived method, the CBiRRT2 algorithm [8] uses projection and rejection sampling to extend a tree from the starting and the goal configurations on the constraint manifold, until finding the connection between both trees, which provides the required solution. In our experiments we guide the search of the RRT and the CBiRRT2 algorithm with learned sampling distributions.

To gather experience before the query, a pre-computed discrete roadmap (using PRM) allows the identification of relevant connections in the C-space. Lazy PRM [9] delays collision checks to gain efficiency in slightly changing environments. Experience Graphs [10] construct a discrete graph from previous paths, which guides the current query based on heuristics. Similarly, Experience-Based Planning with Sparse Roadmap Spanners [11] constructs a sparse roadmap and gains new experience with a parallel RRT planner. In our previous work on Repetition Roadmaps (RepMaps) [5], we constructed a roadmap by connecting the individual components of a learned distribution, which creates a roadmap that is able to capture the relevant portion of the C-space.

Adaptive sampling-based approaches encode the experience from previous solutions in the sampling distribution of the planner. The resulting planners are not bound to discrete states, but contain less connectivity information of the C-space than roadmap methods. Existing methods learn sampling distributions, for example with reinforcement learning based on workspace features [12]. A recent trend investigates density estimation of sampling distributions in the C-space of the robot based on previous experience. These sampling distributions are either constructed with Kernel Density Estimation [13], trained with conditional variational auto encoders [1, 14] or generative adversarial networks [15, 3]. None of these methods attempts to transfer the sampling distribution in between robots with different kinematic structures. The method here proposed computes sampling distributions based on our previous work with Gaussian Mixture Models [4], but any of the mentioned methods could potentially be used for the density estimation.

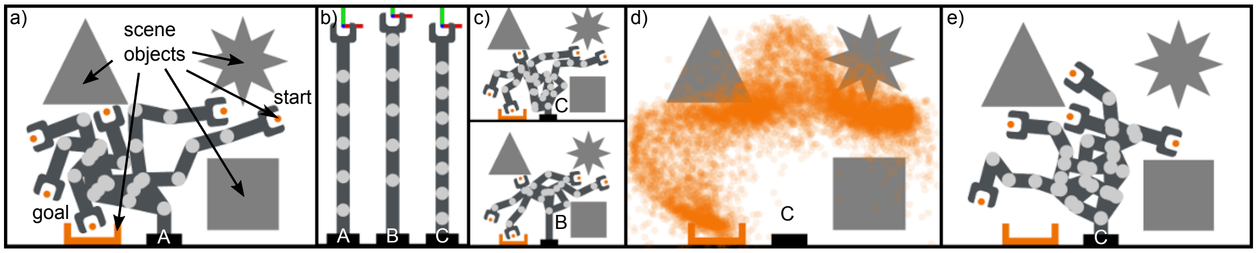

Figure 2: Overview of the transfer learning process with planar manipulators. a) shows the repetitive motion planning problem: all objects as well as the start configuration vary between problem instances. b) shows three different kinematics with two 5 DoF (A, B), and a 7 DoF (C) planar manipulators. c) shows the translated path of kinematic A from a) into kinematics B and C. d) visualizes the learned sampling distribution transferred from 200 solution paths from A to C, by sampling 10000 configurations and overlaying the end-effector frames as dots. e) shows five configurations, sampled from the distribution.

3 KINEMATIC TRANSFER PROBLEM

In this work we design an algorithm to solve a *repetitive* motion planning problem. The robot has a certain start configuration $\mathbf{q}_\mathrm{s}$ and is requested to move the tool or grasped object to a certain goal pose T_g relative to the robot base, following a solution path $\mathcal{P}$ of configurations $\mathbf{q}_\mathrm{i}$. In each instance of the problem the same objects are in the scene but each object has a locally different pose, assumed to be normally distributed. Not all objects have to be present at each problem instance, but the algorithm performs best when each object is present at some point during the learning phase. In each problem instance, the manipulator starts at a different $\mathbf{q}_\mathrm{s}$ and can plan to a different pose T_g.

In addition, there are M previous solution paths $\mathcal{P}_{\mathrm{m,S}}$ available in the C-space of a source robot with a similar workspace, but with a different kinematic structure than the current target robot. The goal of the algorithm is to exploit the information of these paths to efficiently solve new instances of the repetitive motion planning problem.

4 KINEMATIC TRANSFER LEARNING

The basic idea of the algorithm is to augment the sampling distribution of a sampling-based planner to efficiently sample the relevant C-space for the repetitive motion planning problem at hand. By focusing the search on the relevant C-space, the algorithm saves computation time.

To estimate a sampling distribution of a target robot from the solutions of a different source robot, the algorithm first A) transfers the configurations into the C-space of the target robot. Based on the transferred configurations the algorithm B) estimates a sampling distribution in the C-space of the target robot. During production, the algorithm exploits the sampling distribution to C) guide the sampling-based planner with samples drawn from the distribution.

4.1 Transfer solution paths

To transfer the configurations $\mathbf{q}_{\mathrm{i,S}}$ of the M previous solutions $\mathcal{P}_{\mathrm{m,S}}$ from the source robot to the target robot, we assume that the end-effector poses of the source robot are relevant for the planning problem of the target robot. This is a reasonable assumption: To solve the same task in a similar environment, both end-effectors follow similar motions.

The algorithm transfers each configuration $\mathbf{q}_{\mathrm{i,S}}$ of a solution path $\mathcal{P}_{\mathrm{m,S}}$ of the source manipulator into L configurations $\mathbf{q}_{\mathrm{i,T}}$ of the target manipulator. For each $\mathbf{q}_{\mathrm{i,S}}$ the algorithm computes the pose of a specific transfer frame $T_{\mathrm{T,S}}$ of the source manipulator. An inverse-kinematics (IK) solver computes the L configurations for a matching transfer frame $T_{\mathrm{T,T}}$ on the target manipulator. The IK query can result in no solution or infinitely many solutions. In the former case, we skip the configuration $\mathbf{q}_{\mathrm{i,S}}$ as it is not important to capture every configuration for the sampling distribution. To avoid the case of more than L solutions, the algorithm enforces a minimum joint space distance d between different configurations, and selects L

configurations randomly. The IK solver can be an analytic solver for manipulators with known analytic equation (fast), or a solver that relies on projection sampling, i.e., sampling configurations and following a gradient-descent to the desired pose (slow). Figures 2 a) and 2 c) show a path transferred from a planar manipulator with kinematic A to the kinematics of B and C using the transfer frames shown in Figure 2 b).

The algorithm iterates through every configuration $\mathbf{q}_{\mathrm{i,S}}$ of each solution path $\mathcal{P}_{\mathrm{m,S}}$ and collects the N transferred configurations $\mathbf{q}_{\mathrm{i,T}}$ in a learning set $\mathbf{Q}$ - the basis for estimating the sampling distribution for the target manipulator.

4.2 Estimating the sampling distribution

Once the configurations are transferred into the C-space of the target manipulator, we can apply the repetition sampling algorithm we previously published [4]. The algorithm estimates a Gaussian Mixture Model (GMM) from the learning set $\mathbf{Q}$, which contains the stacked N configurations $\mathbf{q}_{\mathrm{i,T}}$:

$$\mathbf{Q} = \begin{bmatrix} \mathbf{q}_{\mathrm{0,T}} & \dots & \mathbf{q}_{\mathrm{i,T}} & \dots & \mathbf{q}_{\mathrm{N,T}} \end{bmatrix}^T \tag{1}$$

The algorithm starts by initializing the distribution with k-Means and estimates the GMM using Expectation Maximation (EM) [16]. The algorithm iteratively computes the responsibility $r_{\mathrm{i,k}}$ of each Gaussian $\mathcal{N}_{\mathrm{k}}$ representing the configuration $\mathbf{q}_{\mathrm{i,T}}$ as

$$r_{\mathrm{i,k}} = \frac{w_{\mathrm{k}}\mathcal{N}_{\mathrm{k}}(\mathbf{q}_{\mathrm{i,T}})}{\sum_{j=1}^{K} w_{\mathrm{j}}\mathcal{N}_{\mathrm{j}}(\mathbf{q}_{\mathrm{i,T}})} \tag{2}$$

Based on the responsibility the EM algorithm recomputes the mean of each component

$$\mu_{\mathrm{k}} = \frac{1}{R_k}\sum_{i=1}^{N} r_{\mathrm{i,k}}\mathbf{q}_{\mathrm{i,T}}, \tag{3}$$

the variance of each component

$$\Sigma_{\mathrm{k}} = \frac{1}{R_k}\sum_{i=1}^{N} r_{\mathrm{i,k}}(\mathbf{q}_{\mathrm{i,T}} - \mu_{\mathrm{k}})(\mathbf{q}_{\mathrm{i,T}} - \mu_{\mathrm{k}})^T, \tag{4}$$

and the new weight of each component

$$w_{\mathrm{k}} = \frac{R_k}{N}, \tag{5}$$

where R_k represents the sum of responsibilities of this component.

$$R_k = \sum_{i=1}^{N} r_{\mathrm{i,k}}, \tag{6}$$

The EM algorithm iterates until the log likelihood

$$\ln p(\mathbf{Q}) = \sum_{i=1}^{N} \ln\left(\sum_{k=1}^{K} r_{\mathrm{i,k}}\mathcal{N}_k(\mathbf{q}_{\mathrm{i,T}})\right) \tag{7}$$

reaches a local optimum.

4.3 Guiding the sampling-based planner

During production, the algorithm guides the sampling-based planner with the new sampling distribution into the relevant C-space. During each expansion step of the planner, new samples are drawn from the learned distribution instead of using a uniform distribution. In our experiments we bias the sampling distribution of a Bi-directional RRT (BiRRT) planner for the planar manipulator experiment and a Constrained Bi-directional RRT (CBiRRT2) [8] for the cobot experiments. Nevertheless the distributions can guide any sampling-based-planner, for example PRM-based methods.

5 EXPERIMENTS

We evaluated the presented algorithm in two experiments: A 2D planar toy example to gain insights and visualize the distributions, and a constrained manipulation experiment in which existing collaborative robots pick and store a container without spilling its contents. The statistics for both experiments were computed on a desktop computer with an Intel(R) i7 CPU with 2.8 GHz and 16 GB of RAM.

5.1 Planar manipulators

An overview of the planar planning problem is depicted in Figure 2 a). The manipulator delivers the orange object to the goal, without colliding with the three gray obstacles. In each problem instance the poses of all objects vary locally and are sampled from a normal distribution each. We created three manipulators depicted with the chosen transfer frames in Figure 2 b). The structure for robots A and B both consist of five revolute joints placed at different positions, while kinematic C consists of seven revolute joints.

We recorded a learning set of solutions on 200 problem instances for each of the three manipulators using a Bidirectional RRT (BiRRT) with uniform sampling. To decrease the learning set path length we employed a shortcut algorithm for 100 ms on each solution [17]. From the learning set we estimated nine distributions, one for each tuple of kinematics. We biased the BiRRT with each sampling distribution and evaluated each resulting planner with 50 test problem instances. The resulting data are depicted in Figures 4 and 5, as well as in Table 1.

Figure 4 shows that the mean number of collision checks is significantly lower for all planners with a learned sampling distribution than for the planners with a uniform distribution. The reduction ranges from a factor of 2.2, for the case of the 5 DoF A kinematic trained on the 7 DoF C kinematic, to a factor of 9.2, for the case of the 7 DoF C kinematic trained on the 5 DoF B kinematic. Figure 5 shows that this trend is not as pronounced in the computation time speedup, which ranges from a factor of 1.1 to 3.4. We investigated the statistical relevance in a two-sided Wilcoxon test: Comparing the number of collision checks of each learned distribution with the uniform distribution, for each target robot, yields a maximum p-value of 2.1×10^{-12}, indicating high significance of the results.

5.2 Collaborative Robots

We evaluated the algorithm in a realistic setting using five collaborative robots (cobots). As depicted in Figure 3, the task of the cobot is to move a container from the table into a bin. To not spill any contents, the motion has to respect an additional orientation constraint. In each problem instance the poses of the bin and container vary locally, and one of the two deposit locations in the bin is already occupied by another container. In the target application the operator can repeatedly place the relevant objects in the same general poses on the table, but does not need to precisely position the objects. We choose five existing cobots shown with their transfer frames in Figure 3 b): The DLR SARA robot (7 DoF), the KUKA IIWA robot (7 DoF), the Universal UR10E (6 DoF), the Franka PANDA robot (7 DoF), and the Rethink SAWYER robot (8 DoF - 7 relevant for the task). For all robots we placed the transfer frames onto the inner palm of their gripper, keeping the same orientation.

We recorded a learning set of solutions on 500 problem instances for each of the cobots using a CBiRRT2 planner with uniform sampling. To decrease the learning set path length we a employed a shortcut algorithm for 100 ms on each solution [17]. From the learning set we estimated 25 distributions, one for each tuple of kinematics.

For each distribution an average of 72160 configurations were transferred in an average time of 6.52 s with analytic inverse kinematics solvers generated with IKFast [18], and the EM algorithm to compute each distribution took an average time of 42.3 s. We biased the CBiRRT2 with each sampling distribution and evaluated each resulting planner with 100 test problem instances. The resulting data is depicted in Figures 6 and 7, as well as in Table 2.

Figure 6 shows that the mean number of collision checks is significantly lower for all planners with a learned sampling distribution than for the planners with a uniform distribution. The reduction in collision check ranges from a factor of 1.4, for the case of the SARA trained on the UR10E robot, to a factor of 3.9, for the case of the UR10E trained on the IIWA robot. Figure 5 shows that this trend is similar in the computation

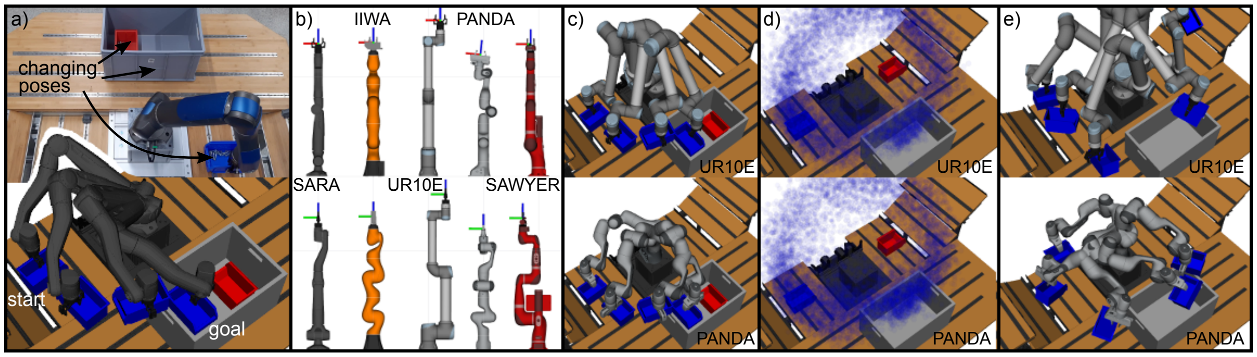

Figure 3: Overview of the transfer learning experiment with manipulators. a) shows the repetitive motion planning problem: all objects as well as the start configuration vary between problem instances. b) depicts two views of the five cobots used in the experiments, as well as the used transfer frame. c) shows the translated paths of the UR10E and the PANDA robot from the path of the SARA robot in a). d) visualizes the learned sampling distributions transferred from 500 solution paths from SARA to UR10E and PANDA, by sampling 10000 configurations and overlaying the end-effector frames as dots. e) shows five configurations each, sampled from the same distribution.

time: The speedup ranges from a factor of 1.3 to a factor of 3.9. We investigated the statistical relevance in a two-sided Wilcoxon test: Comparing the computation time of each learned distribution with the uniform distribution, for each target robot, yields a maximum p-value of 9.4×10^{-4}, indicating high significance of the results. Table 2 shows a slight trend that the learned sampling distributions produce shorter paths, but no statistically relevant claim can be made.

In addition to the results from simulated experiments, we executed a few of the computed motions on the SARA robot to solve the task with different object placements. The motions can be seen in the accompanying video material.

6 DISCUSSION

The data from the experiments suggest that transferring experience from one robot increases the efficiency of the search for a new motion of a target robot. Both experiments show significantly reduced collision checks for all motion planners that were guided by the learned sampling distributions, when compared to uniform sampling. This indicates that the learned sampling distribution focuses the search onto the relevant C-space, which leads to significant computation time speedups in the cobot experiment. In addition, the variance in computation time is significantly lower, increasing the predictability of the planning cycle time.

The reduction in computation time seems to correlate to how complex the motion is to compute for a certain robot. The robots with the highest number of mean collision checks with uniform sampling, UR10E as well as SAWYER, also have the highest computation time speedups. The SARA robot with the lowest amount of collision checks during uniform sampling, has the lowest computation time speedup.

The average computation time of 6.52 s for transferring the configurations from the source manipulator to the target manipulator shows the computational benefit, instead of computing a new learning set for the target manipulator, which would take at least 395 s for the best case of the IIWA manipulator. Re-computation of the learning set can also be difficult in cases where the obstacles are not available as meshes, but for example perceived live from RGB-D data.

The data show no significant difference in path lengths between uniform sampling and the learned sampling distributions, although the learning set solutions were locally optimized. This indicates that it is difficult to transfer path qualities between the robots, since the C-spaces of the robots are fundamentally different.

Surprisingly, the best computation time for each robot is not necessarily achieved with the sampling distribution trained on the robot itself, as seen for the SARA and UR10E robots. One potential explanation is that the learning sets of both robots contained less relevant information. Increasing the learning set size might help alleviate this problem.

In the experiments we investigated serial manipulators with revolute joints (5R, 6R, 7R), targeting kinematics of most cobots today [19]. Future work could investigate the transferrability to different kinematic types, including parallel kinematics or kinematics with prismatic joints, and derive a transferrability measure in-between different kinematics.

7 CONCLUSIONS

Transferring the experience of one manipulator to a new manipulator in the same application domain can decrease the computation time for motion planning significantly. The proposed algorithm for transferring configurations and estimating a sampling distribution for a sampling-based planner can achieve this goal, as shown by the data presented in the experiments.

References

[1] B. Ichter, J. Harrison, and M. Pavone, "Learning sampling distributions for robot motion planning," in *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 7087–7094, 2018.

[2] A. H. Qureshi, Y. Miao, A. Simeonov, and M. C. Yip, "Motion planning networks: Bridging the gap between learning-based and classical motion planners," *IEEE Transactions on Robotics*, vol. 37, no. 1, pp. 48–66, 2021.

[3] T. S. Lembono, E. Pignat, J. Jankowski, and S. Calinon, "Learning constrained distributions of robot configurations with generative adversarial network," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 4233–4240, 2021.

[4] P. Lehner and A. Albu-Schäffer, "Repetition sampling for efficiently planning similar constrained manipulation tasks," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pp. 2851–2856, 2017.

[5] P. Lehner and A. Albu-Schäffer, "The repetition roadmap for repetitive constrained motion planning," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 3884–3891, 2018.

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

[7] S. M. LaValle, "Rapidly exploring dense trees," *Planning Algorithms*, pp. 228–237, 2004.

[8] D. Berenson, S. S. Srinivasa, and J. Kuffner, "Task space regions: A framework for pose-constrained manipulation planning," *The Int. Journal of Robotics Research*, vol. 30, no. 12, pp. 1435–1460, 2011.

[9] R. Bohlin and L. E. Kavraki, "Path planning using lazy PRM," in *Proc. IEEE Int. Conf. Robotics and Automation*, vol. 1, pp. 521–528, 2000.

[10] M. Phillips, B. J. Cohen, S. Chitta, and M. Likhachev, "E-graphs: Bootstrapping planning with experience graphs.," in *Robotics: Science and Systems VIII*, pp. 337–344, 2012.

[11] D. Coleman, I. A. Şucan, M. Moll, K. Okada, and N. Correll, "Experience-based planning with sparse roadmap spanners," in *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 900–905, 2015.

[12] M. Zucker, J. Kuffner, and J. A. Bagnell, "Adaptive workspace biasing for sampling-based planners," in *Proc. IEEE Int. Conf. Robotics and Automation*, pp. 3757–3762, 2008.

[13] T. F. Iversen and L.-P. Ellekilde, "Kernel density estimation based self-learning sampling strategy for motion planning of repetitive tasks," in *Proc. IEEE Int. Conf. Intelligent Robots and Systems*, pp. 1380–1387, 2016.

[14] D. Molina, K. Kumar, and S. Srivastava, "Learn and link: Learning critical regions for efficient planning," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, pp. 10605–10611, 2020.

[15] R. Gieselmann and F. T. Pokorny, "Standard deep generative models for density estimation in configuration spaces: A study of benefits, limits and challenges," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pp. 5238–5245, 2020.

[16] C. Bishop, "Mixture models and EM," *Pattern Recognition and Machine Learning*, pp. 423–455, 2006.

[17] R. Geraerts and M. H. Overmars, "Creating high-quality paths for motion planning," *The Int. Journal of Robotics Research*, vol. 26, no. 8, pp. 845–863, 2007.

[18] R. Diankov, "Inverse kinematics," *Automated Construction of Robotic Manipulation Programs, Ph. D. dissertation*, pp. 78–98, 2010.

[19] E. Matheson, R. Minto, E. G. Zampieri, M. Faccio, and G. Rosati, "Human–robot collaboration in manufacturing applications: a review," *Robotics*, vol. 8, no. 4, p. 100, 2019.

Table 1: Results for the planar manipulators experiment

Evaluated Robot / RRT Guidance	5 DoF A	5 DoF B	7 DoF C
Computation time mean [s]			
Uniform	0.42	0.49	1.57
5 DoF A	0.38	0.32	0.80
5 DoF B	0.30	0.30	0.46
7 DoF C	0.38	0.34	0.78
Computation time std. dev. [s]			
Uniform	0.20	0.27	0.78
5 DoF A	0.21	0.15	0.59
5 DoF B	0.17	0.14	0.25
7 DoF C	0.26	0.16	0.54
Collision checks mean [#]			
Uniform	5176	4695	20354
5 DoF A	2356	1862	3940
5 DoF B	1863	1772	2204
7 DoF C	2367	2054	3786
Collision checks std. dev. [#]			
Uniform	2830	2867	10692
5 DoF A	1409	952	2971
5 DoF B	1128	871	1229
7 DoF C	1669	977	2640
Path length (joint distance) mean [deg]			
Uniform	758.90	890.40	968.10
5 DoF A	732.90	848.60	897.40
5 DoF B	777.50	880.00	873.90
7 DoF C	747.20	886.90	882.80
Path length (joint distance) std. dev. [deg]			
Uniform	192.40	220.40	280.50
5 DoF A	187.00	199.60	223.70
5 DoF B	194.00	226.30	224.90
7 DoF C	212.80	218.70	246.60

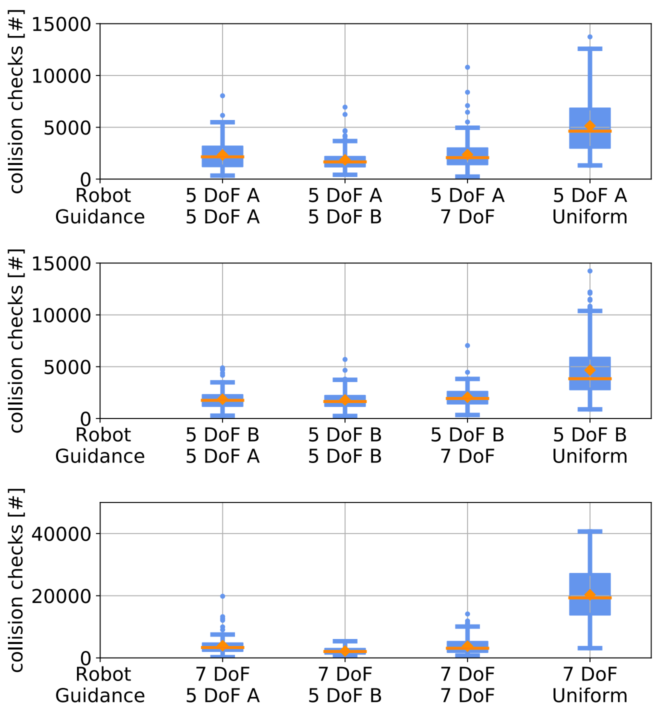

Figure 4: Barplots of the number of collision checks in the planar manipulator experiment with the manipulators shown in Figure 2 b). Guidance denotes the source robot from which the sampling distribution was derived, while Robot denotes the target robot. Uniform indicates uniform sampling.

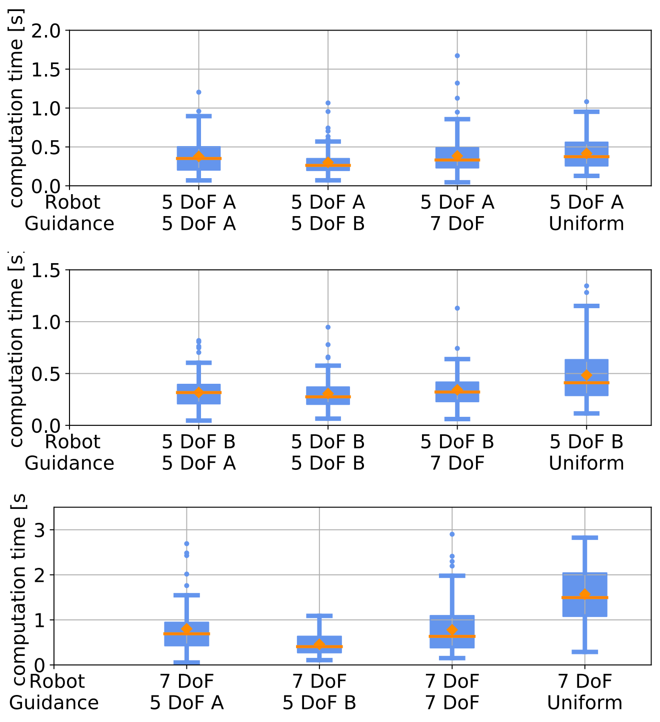

Figure 5: Barplots of the computation time in the planar manipulator experiment with the manipulators shown in Figure 2 b). Guidance denotes the source robot from which the sampling distribution was derived, while Robot denotes the target robot. Uniform indicates uniform sampling.

Table 2: Results for the cobot experiment

Evaluated Robot / RRT Guidance	SARA	IIWA	UR10E	PANDA	SAWYER
Computation time mean [s]					
Uniform	0.80	0.79	2.51	1.08	2.59
SARA	0.59	0.38	1.22	0.69	1.50
IIWA	0.42	0.32	0.64	0.49	1.23
UR10E	0.61	0.38	0.67	0.39	1.39
PANDA	0.44	0.36	0.80	0.35	1.22
SAWYER	0.45	0.38	0.78	0.44	1.19
Computation time std. dev. [s]					
Uniform	0.73	0.73	2.51	0.96	2.34
SARA	0.50	0.29	0.99	0.64	1.18
IIWA	0.34	0.24	0.52	0.51	1.06
UR10E	1.96	0.34	0.56	0.45	1.15
PANDA	0.34	0.27	0.71	0.33	0.90
SAWYER	0.36	0.32	0.59	0.39	0.97
Collision checks mean [#]					
Uniform	5689	8054	11604	10321	9155
SARA	3667	3602	4763	5678	4802
IIWA	2827	2959	2707	4109	4183
UR10E	3959	3519	2971	3044	4514
PANDA	2793	3221	3276	2824	3931
SAWYER	3078	3549	3339	3693	4044
Collision checks std. dev. [#]					
Uniform	5114	7313	11421	8984	8230
SARA	3089	2733	3790	5076	3732
IIWA	2242	2157	2166	4178	3557
UR10E	12557	3032	2518	3375	3690
PANDA	2088	2289	2888	2579	2872
SAWYER	2447	2891	2560	3221	3284
Path length (joint distance) mean [deg]					
Uniform	880.99	947.69	1065.13	884.91	1038.26
SARA	816.39	846.28	791.32	679.50	909.09
IIWA	819.79	781.57	776.25	643.63	858.48
UR10E	866.78	819.03	802.66	637.73	855.76
PANDA	770.50	757.46	788.95	565.99	882.37
SAWYER	834.94	777.53	755.48	634.27	949.92
Path length (joint distance) std. dev. [deg]					
Uniform	388.82	457.51	419.81	392.87	454.90
SARA	418.01	412.26	348.72	297.53	437.16
IIWA	420.43	402.76	339.86	336.45	387.98
UR10E	425.52	396.06	331.10	341.34	398.10
PANDA	385.64	377.02	346.76	300.37	412.89
SAWYER	398.78	373.37	328.53	329.80	400.35

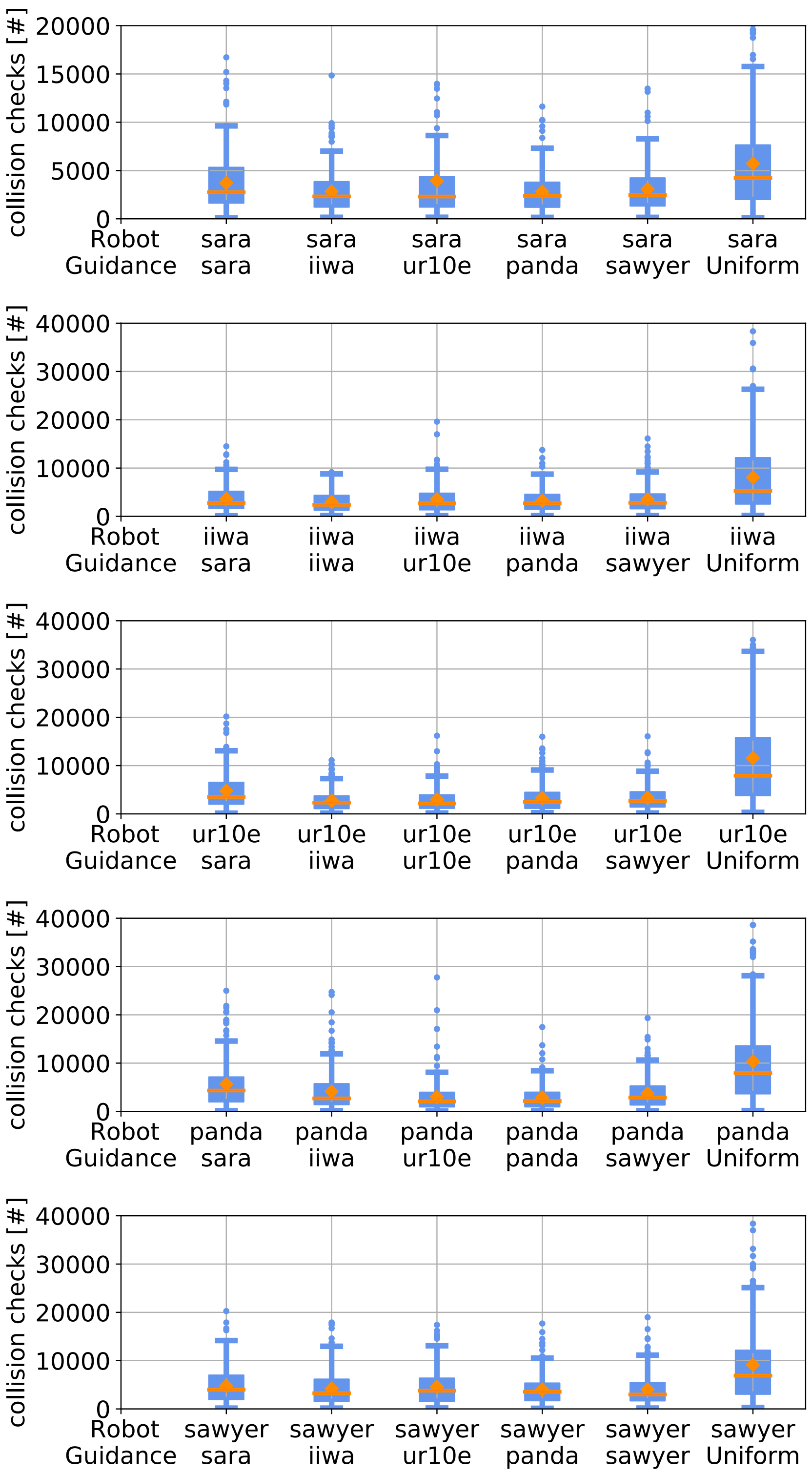

Figure 6: Barplots of the number of collision checks in the cobot manipulator experiment with the manipulators shown in Figure 3 b). Guidance denotes the source robot from which the sampling distribution was derived, while Robot denotes the target robot. Uniform indicates uniform sampling.

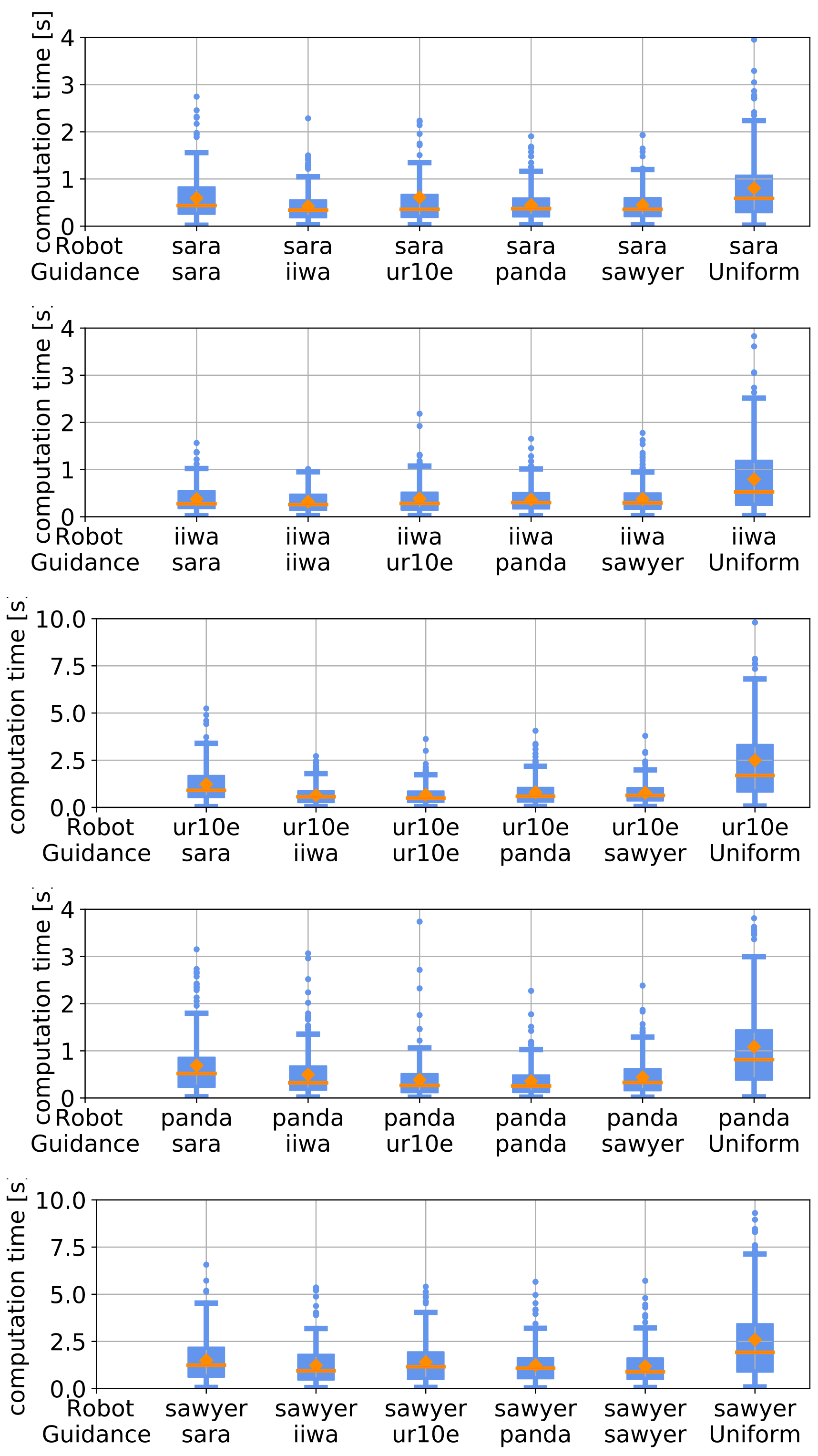

Figure 7: Barplots of the computation time in the cobot manipulator experiment with the manipulators shown in Figure 3 b). Guidance denotes the source robot from which the sampling distribution was derived, while Robot denotes the target robot. Uniform indicates uniform sampling.