Fast Deep Swept Volume Estimator

Journal Title
XX(X):1–16

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Hao-Tien Lewis Chiang[1,2], John E. G. Baxter[1], Satomi Sugaya[1], Mohammad R. Yousefi[1], Aleksandra Faust[2], Lydia Tapia[1]

Abstract
Despite decades of research on efficient swept volume computation for robotics, computing the exact swept volume is intractable and approximate swept volume algorithms have been computationally prohibitive for applications such as motion and task planning. In this work, we employ Deep Neural Networks (DNNs) for fast swept volume estimation. Since swept volume is a property of robot kinematics, a DNN can be trained off-line once in a supervised manner and deployed in any environment. The trained DNN is fast during on-line swept volume geometry or size inferences. Results show that DNNs can accurately and rapidly estimate swept volumes caused by rotational, translational and prismatic joint motions. Sampling-based planners using the learned distance are up to 5x more efficient and identify paths with smaller swept volumes on simulated and physical robots. Results also show that swept volume geometry estimation with a DNN is over 98.9% accurate and 1200x faster than an octree-based swept volume algorithm.

Keywords
Swept Volume, Motion Planning, Deep Learning

1 Introduction

Some of the earliest work in robotic motion planning noted the importance of swept volume computation since swept volume, $\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)$, is the physical volume displaced by an object moving along a trajectory between two configurations, $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ (Figure 1a). For example, early motion planning work in configuration space (c-space) proposed to use swept volume to construct c-space obstacle surfaces (Lozano-Perez 1990). Moreover, core components of modern sampling-based motion planning, i.e., distance measures between high Degree Of Freedom (DOF) configurations and continuous collision detection, can also benefit significantly from the use of swept volumes (Kuffner 2004; Xavier 1997). In fact, the size of swept volume, or swept volume measure ($|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$) was identified as the ideal distance measure, since it can reflect the collision probability between configurations [*]. As a result, using the swept volume as the distance measure reduces the number of unsuccessful local planning attempts. (Kuffner 2004). Lastly, swept volumes can be used as an efficient bridge between task and motion planning (Gaschler et al. 2013; Kaelbling and Lozano-Pérez 2010).

Unfortunately, exact swept volume computation is generally intractable due to the complex non-linear relation between the configuration pair (for start and end poses) and corresponding geometry. As a result, most algorithms focus on generating approximations (Kim et al. 2004), such as occupation grid-based, convex polyhedra-based and boundary-based methods (Kim et al. 2004; Himmelstein et al. 2010; Von Dziegielewski et al. 2015; Gaschler et al. 2013; Campen and Kobbelt 2010; Abrams and Allen 2000). Worse, despite more than four decades of study, approximate swept volume algorithms can still be inadequate for applications such as motion planning due to being overly conservative or having high computational costs (Gaschler et al. 2013; Ekenna et al. 2015).

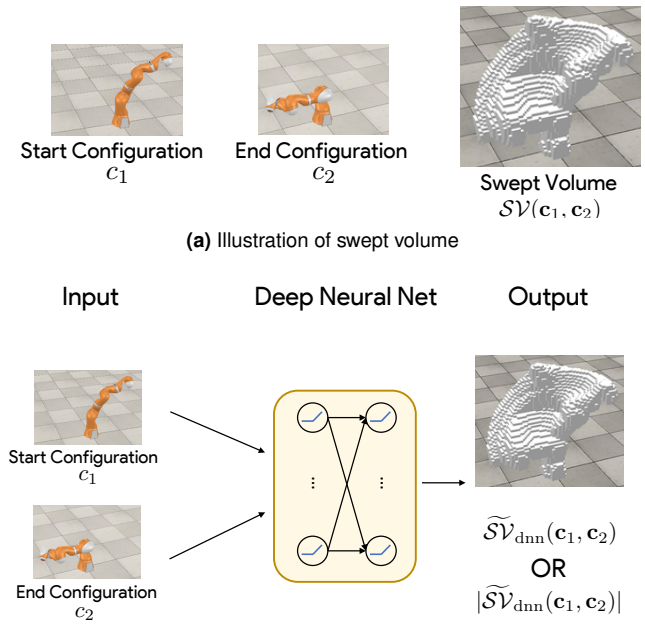
To provide fast swept volume estimation, we propose using Deep Neural Networks (DNNs) to approximate swept volumes (Figure 1b). Trained off-line with swept volume examples for a given robot in an obstacle-free space, the DNN captures the complex and nonlinear relationship between trajectories in the robot's configuration space. As a result, DNNs significantly reduce computation costs during on-line swept volume estimations. Successful learning of swept volume is feasible since DNNs can approximate any continuous bounded function (Hornik 1991) and we show that swept volume possesses these properties in a finite configuration space.

The primary contributions are as follows. First, we develop techniques for using fast swept volume measure ($|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$) estimation with DNN as a distance measure to improve sampling-based motion planner performance. Second, we briefly demonstrate that swept volume geometries ($\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)$) can be accurately and efficiently estimated by DNNs. This paper extends our previous work (Chiang et al. 2018) through the following additional contributions. First,

[1]University of New Mexico, USA
[2]Robotics at Google, USA

Corresponding author:
Lydia Tapia, Computer Science, University of New Mexico, Albuquerque, New Mexico, USA

[*] $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ is a measure that has been traditionally used as a proxy for preferring paths that displace the least volume. When obstacles are uniformly randomly located and without any priors, paths with the smallest $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ have the least probability of collision.



(a) Illustration of swept volume

(b) Fast swept volume estimation with deep neural nets

Figure 1. (a) Illustration of swept volume and (b) fast swept volume estimation with DNNs. In this work, we assume the trajectory connecting $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ is a straight line in the configuration space unless otherwise specified.

we comprehensively evaluate the accuracy of DNN distance measure learning on 7 robots, encompassing rigid body, prismatic and revolute joints and closed-loop kinematic chain motion.

Next, we tested the learned distance measure on sampling-based planners, PRM (Kavraki et al. 1996), RRT (LaValle and Kuffner 2001) and RRT-Connect (Kuffner and LaValle 2000), in four simulated motion planning scenarios and one scenario on a physical Baxter robot. Lastly, we discuss the trade-offs between distance measures and demonstrate accurate swept volume geometry estimation. This paper might also be of interest to the larger robotics, computational geometry and computer graphics communities, particularly in the area of applying machine learning techniques to high degree of freedom systems.

Experimental results indicate that DNN can accurately and rapidly estimate $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ across all 7 robots tested. Planners that use the learned distance measure are: 1) up to five times more likely to identify a collision-free path on a fixed time budget and 2) able to return paths with a smaller swept volume. These advantages are consistent for all robots tested and are particularly significant for more complex robots with a highly articulated body.

Results also indicate that swept volume geometry estimation of a 7 DOF manipulator is over 98.9% accurate and up to 1200x faster than a state of the art swept volume algorithm.

2 Related Work

Modern approximate swept volume algorithms can be roughly classified as occupation grid-based, convex polyhedra-based, and boundary-based. Occupation grid-based approaches decompose the workspace, e.g., into voxels, in order to record the robot's occupation in the workspace as it executes a trajectory (Himmelstein et al. 2010; Von Dziegielewski et al. 2015). The resulting approximation has a resolution-tunable accuracy and is conservative, which can be critical for applications such as collision avoidance (Perrin et al. 2012).

Convex polyhedra-based methods approximate robot bodies as the union of simple convex hulls using algorithms such as (Mamou and Ghorbel 2009). As the robot moves between configurations, additional convex hulls are inserted at a fixed angular interval and the swept volume is the union of all convex hulls (Gaschler et al. 2013). The boundary-based methods extract the boundary surface (Campen and Kobbelt 2010; Kim et al. 2004; Abrams and Allen 2000). Despite more than four decades of study, swept volume computation is still too slow to be used on-line by sampling-based motion planners (Kuffner 2004; Ekenna et al. 2015).

Swept volume has been used in motion planning in various ways. In (Perrin et al. 2012), swept volumes of enumerated step patterns of a bipedal robot are computed off-line and queried on-line by an RRT-based planner to speed up collision detection for robot footstep planning. Similarly, in this paper we compute swept volume approximations off-line. However, our learned estimators can generalize to unseen configuration pairs. Swept volume has also been used directly as a distance measure in (Ekenna et al. 2015). However, due to the exceedingly high swept volume computation cost, the performance is reported to be orders of magnitude worse than weighted Euclidean distance measure.

A distance metric that accurately predicts a local planner's success rate is critical for sampling-based motion planners (Elbanhawi and Simic 2014). On the other hand, distance metric calculations also need to be fast since they are one of the most numerous sampling-based planner operations (Amato et al. 1998). Carefully tuned weighted Euclidean distance metrics have been empirically shown to outperform other metrics (Amato et al. 1998). This conclusion is echoed in (Völz and Graichen 2016) where the swept volume is approximated by tuning a weighted Euclidean distance measure. However, a weighted Euclidean distance metric may not be expressive enough to approximate swept volume as swept volumes are nonlinear, i.e., each joint DOF affects one another in an articulated body.

Machine learning has been used to learn distance measures for kinodynamic robots (Palmieri and Arras 2015; Wolfslag et al. 2018). In such systems, the design goal of distance metrics is often quite different from planning in configuration space due to the constrained reachability. Good distance metrics typically approximate the minimum time to reach between states (Palmieri and Arras 2015). For example, regression learning algorithms are trained off-line in (Palmieri and Arras 2015) to approximate the time to reach between states. The training data is generated by a near-optimal controller for unicycle robots. An RRT-based planner then uses the learned distance metric during on-line planning. A similar method replaces the near-optimal controller with an indirect controller to learn both the time to reach and control inputs (Wolfslag et al. 2018). These methods differ from ours, in that our method identifies neighboring configurations that are likely to succeed in the connect or extend operations due to expected distance

of a trajectory, while distance metrics in (Wolfslag et al. 2018; Palmieri and Arras 2015) approximate minimum time to reach. Another example of integration between deep machine learning and sampling-based planning is PRM-RL (Faust et al. 2018). Similar to our method, it uses an off-line, once-per robot training to connect PRM nodes that are most likely to succeed. Unlike our learned distance estimators, PRM-RL learns the local planner for a physical robot moving via sensor information. Therefore, nodes that are most likely to succeed are connected.

3 Problem Formulation

In this paper, unless otherwise specified, we define the swept volume, $\mathcal{SV}$, for a trajectory in configuration space as

$$\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \cup_{t\in[0,1]} \mathcal{V}((1-t)\boldsymbol{c}_1 + t\boldsymbol{c}_2), \tag{1}$$

where $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathbb{R}^{d_f}$ are the start and end configurations of a robot with d_f DOF, and $\mathcal{V}(\boldsymbol{c})$ is the workspace occupied by the robot in configuration $\boldsymbol{c}$. We consider the trajectory between $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ to be a *straight line* in configuration space unless otherwise specified. $\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ can be highly complex and nonlinear due to rotational DOFs, especially in cases where the robot has an articulated body.

Our goal is to train a swept volume measure estimator $\mathcal{D}^{\text{sv}}_{\text{dnn}}(\boldsymbol{c}_1, \boldsymbol{c}_2|\boldsymbol{\theta})$ that approximates the swept volume measure $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ by finding parameters $\boldsymbol{\theta}$ that minimizes loss $\mathcal{L}$:

$$\tilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}\big(\mathcal{D}^{\text{sv}}_{\text{dnn}}(\boldsymbol{c}_1, \boldsymbol{c}_2|\boldsymbol{\theta}), |\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|\big). \tag{2}$$

In addition, as a proof of concept, we also demonstrate learning of a swept volume geometry estimator $\mathcal{SV}_{\text{dnn}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ that approximates $\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)$.

4 Methods

In this section, we describe how to train the swept volume measure estimator with DNNs ($\mathcal{D}^{\text{sv}}_{\text{dnn}}$) and techniques to use it as a distance measure for sampling-based motion planning. The estimator models are trained off-line in two steps. First, $|\mathcal{SV}|$ training dataset generation is described in Section 4.1. Next, using the dataset, we train DNNs in Section 4.2. To overcome the hurdles of utilizing the learned estimator as a distance measure for sampling-based planning, such as the lack of metric space properties and efficiency, we learn an additional weighted Euclidean distance measure, $\mathcal{D}^{\text{sv}}_{\text{we}}$ (Section 4.3). It is then used as a filter in a hierarchical neighbor selector that selects the most promising nodes in sampling-based planning (Section 4.4).

4.1 Training Dataset Generation

The training data of size n is composed of $(\boldsymbol{X}, \boldsymbol{Y})$, where $\boldsymbol{X} = [\boldsymbol{x}_1, \cdots, \boldsymbol{x}_n]^\top$, $\boldsymbol{Y} = [y_1, \cdots, y_n]^\top$. Each training sample $\boldsymbol{x}_i = [\boldsymbol{c}_{1,i}\, \boldsymbol{c}_{2,i}]$ consists of the start and end points uniform-randomly sampled from the configuration space. The ground truth labels, $\boldsymbol{Y}$, match the swept volume measure $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ between two corresponding configurations with respect to the straight line trajectory in c-space. Since the swept volume only relates to the kinematics of the robot and is independent to the environments it operates in, we do not consider obstacles during the generation of training data. Ideally, the labels should be computed with (1), but computing the exact swept volume is intractable. Instead, we approximate the labels with an octree-based swept volume algorithm (Von Dziegielewski et al. 2015).

To approximate the swept volume, the robot trajectory is represented by N_{lerp} intermediate states in c-space. The j^{th} intermediate configuration is

$$\boldsymbol{c}_{j,i} = \left(1 - \frac{j}{N_{\text{lerp}}}\right)\boldsymbol{c}_{1,i} + \frac{j}{N_{\text{lerp}}}\boldsymbol{c}_{2,i}, \;\; j = 1, \cdots, N_{\text{lerp}}. \tag{3}$$

Next, the forward kinematics of the robot maps each $\boldsymbol{c}_{j,i}$ to the workspace occupancy of the robot,

$$\mathcal{G}_j(x, y, z) = \begin{cases} 1, & \text{robot overlaps with } (x, y, z) \in \mathcal{V}(\boldsymbol{c}_{j,i}) \\ 0, & \text{otherwise.} \end{cases}, \tag{4}$$

where x, y, z are the positional coordinates in the 3D workspace, and $\mathcal{V}(\boldsymbol{c}_{j,i})$ is the workspace volume occupied by $\boldsymbol{c}_{j,i}$. The swept volume between $\boldsymbol{c}_{1,i}$ and $\boldsymbol{c}_{2,i}$ can be approximated by taking the union of all $\mathcal{G}_j$,

$$\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \bigcup_{j=1,\cdots,N_{\text{lerp}}} \mathcal{G}_j(x, y, z). \tag{5}$$

The occupancy and the union operation are approximated by an octree decomposition of workspace up to a resolution, Δ, in order to speed up computation compared to a uniform voxel grid of the same resolution.

One undesirable property of using $|\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ as a distance measure for sampling-based planning is that $|\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ is non-zero even when $\boldsymbol{c}_1 \approx \boldsymbol{c}_2$. It roughly equals to the volume of the robot in configuration $\boldsymbol{c}_1$. As a result, $|\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ is biased by the volume at $\boldsymbol{c}_1$. This bias is particularly significant for robots with prismatic joints, where configurations change the size of robot volume. To address this, we subtract the start and end robot workspace occupation from $\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$,

$$\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2) = \widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2) \setminus \mathcal{G}_1(x, y, z) \setminus \mathcal{G}_{N_{\text{lerp}}}(x, y, z), \tag{6}$$

where $\setminus$ represents the set minus operation. As a result, $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}, \boldsymbol{c})| = 0$, $\forall \boldsymbol{c}$. Finally, the i^{th} swept volume measure ground truth label is

$$y_i = |\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|. \tag{7}$$

4.2 Deep Swept Volume Measure Estimator, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$

We use a deep neural network to learn a non-linear swept volume measure model, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$. $\mathcal{D}^{\text{sv}}_{\text{dnn}}((\boldsymbol{c}_1, \boldsymbol{c}_2)|\boldsymbol{\theta})$ is a fully-connected feed-forward DNN parameterized by $\boldsymbol{\theta}$. The inputs, outputs, and the architecture of the $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ are described in Figure 2. The inputs are $2d_f$ input neurons, corresponding to $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$.The N_{hidden} hidden layers consist of ReLu (Hahnloser et al. 2000) neurons. Finally, the output is a neuron estimating the swept volume measure between two configuration points and outputs zero if the network

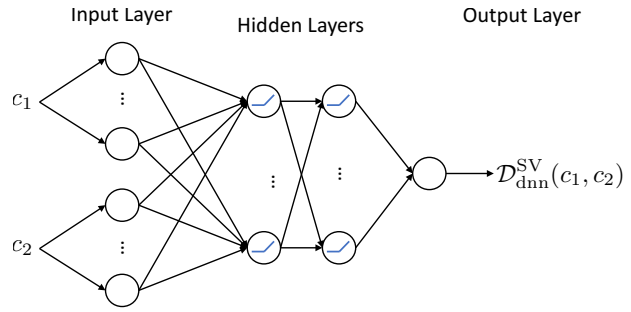


Figure 2. Deep swept volume measure estimator $\mathcal{D}^{\text{sv}}_{\text{dnn}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ neural network architecture used to estimate $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|$. $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are the start and end configurations of the trajectory, respectively. The inputs are $2d_f$. The activation function of the first and last layers are identities. The input layer is connected to the N_{hidden} hidden layers each with N_i ReLU neurons. The output layer has one neuron corresponding to the swept volume measure estimate.

prediction is negative. Stochastic gradient descent backprop finds the weights and biases ($\boldsymbol{\theta}$) w.r.t. the L_2 loss and the training dataset

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left(\mathcal{D}^{\text{sv}}_{\text{dnn}}((\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i})|\boldsymbol{\theta}) - |\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i})|\right)^2. \tag{8}$$

The network is trained once per robot.

4.3 Weighted Euclidean Distance Estimator, $\mathcal{D}^{sv}_{we}$

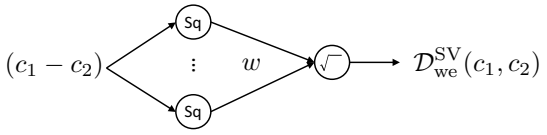


Figure 3. Weighted Euclidean distance estimator, $\mathcal{D}^{\text{sv}}_{\text{we}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$, is a shallow neural net that estimates $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|$. $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ are the start and end configurations of the trajectory, respectively. The input, $\boldsymbol{c}_1 - \boldsymbol{c}_2$, is fed to d_f neurons (one per DOF) with a square activation function, i.e., $f(x) = x^2$. The output is the square-root of absolute value of the weighted sum of activation.

To utilize $\mathcal{D}^{\text{sv}}_{\text{we}}$ efficiently in sampling-based motion planning, we first introduce the weighted Euclidean distance measure, $\text{d}_{\boldsymbol{w}}(\boldsymbol{c}_1, \boldsymbol{c}_2) = \sqrt{\sum_{j=1}^{d_f} w_j (c_{1,j} - c_{2,j})^2}$, which is one of the most commonly used distance measures for sampling-based motion planners (Amato et al. 1998). It often requires manual tuning of the weight vector $\boldsymbol{w} = [w_1, w_2, ..., w_{d_f}] \in \mathbb{R}^{d_f}$. One way to find the weights that best reflect the swept volume measure between two configurations, $\boldsymbol{w}^*$, is to structure the operation as a shallow network, $\mathcal{D}^{\text{sv}}_{\text{we}}$, that models the weighted Euclidean distance measure and approximates the swept volume measure $|\widetilde{\mathcal{SV}}_0|$,

$$\mathcal{D}^{\text{sv}}_{\text{we}}((\boldsymbol{c}_1, \boldsymbol{c}_2)|\boldsymbol{w}^*) = \text{d}_{\boldsymbol{w}*}(\boldsymbol{c}_1, \boldsymbol{c}_2).$$

The details of the network are depicted in Figure 3. $\mathcal{D}^{\text{sv}}_{\text{we}}$ allows us to use a stochastic gradient descent-based optimizer with variable learning rate (Kingma and Ba 2014) to find $\boldsymbol{w}^*$ that minimizes the L_2 loss w.r.t. the training dataset,

$$\boldsymbol{w}^* = \arg min_{\boldsymbol{w}} \sum_{i=1}^{n} \left(\mathcal{D}^{\text{sv}}_{\text{we}}((\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i})|\boldsymbol{w}) - |\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i}|)\right)^2. \tag{9}$$

The network is also trained once per robot, and like the analytical representation of the network, $\mathcal{D}^{\text{sv}}_{\text{we}}$, forms a metric space when the weights are positive.

4.4 Swept Volume-based Hierarchical Neighbor Search, HNS_{sv}

Another hurdle of using $|\widetilde{\mathcal{SV}}_0|$ as a distance measure for sampling-based planning is that $|\widetilde{\mathcal{SV}}_0|$ does not form a metric space as it does not satisfy the triangle inequality. This prevents utilization of efficient nearest neighbor data structures such as GNAT (Brin 1995) that can greatly speedup sampling-based planners (LaValle 2006).

To address this issue, we propose Swept Volume-based Hierarchical Neighbor Search (HNS_{sv}), that combines the trained swept volume measure estimators introduced above ($\mathcal{D}^{\text{sv}}_{\text{dnn}}$ and $\mathcal{D}^{\text{sv}}_{\text{we}}$) to be used for neighbor selection within sampling-based planning. This hierarchical combination efficiently selects neighbors with low swept volume measure, by first filtering all candidates using the extremely fast learned $\mathcal{D}^{\text{sv}}_{\text{we}}$ and then selects the nearest neighbors from this smaller subset with $\mathcal{D}^{\text{sv}}_{\text{dnn}}$. Note that previous hierarchical nearest neighbor selection methods, such as (McMahon et al. 2012), combine Euclidean distance measure and random node selection. Rather, HNS_{sv} is tuned for swept volume estimation. In this paper, we implement HNS_{sv} by using the k-closest neighbor selection method at each level, but other neighbor connection strategies can be used, such as a distance cutoff (Karaman and Frazzoli 2011).

The HNS_{sv} algorithm first identifies k_c candidate nearest neighbors of configuration $\boldsymbol{c}$ using $\mathcal{D}^{\text{sv}}_{\text{we}}$, i.e., the output of the weighted Euclidean distance estimator. This step can be done efficiently by employing efficient nearest neighbor data structures. Next, HNS_{sv} uses the $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ (output of the deep swept volume measure estimator) to choose the final $k_{nn} < k_c$ nearest neighbors among the candidates.

The hierarchical combination of the estimators within neighbor selection has several benefits. First, it enables the use of any efficient nearest neighbor data structure. Second, it greatly reduces the number of $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ queries, which are slower than computing the weighted Euclidean distance. Finally, it selects neighbors with small swept volume measure.

Since HNS_{sv} uses DNN to estimate $|\widetilde{\mathcal{SV}}_0|$, it is difficult to guarantee asymptotic-optimality for planners such as RRT* Karaman and Frazzoli (2011). However, we can maintain the asymptotic-optimality of these planners by simply alternating between HNS_{sv} and traditional connection

strategies based on the Euclidean distance measure in a probabilistic manner. This strategy is similar to the one proposed in Ichter et al. (2018).

5 Continuity of Swept Volume Measure

DNN is a universal approximator for any bounded continuous function (Hornik 1991). In this section we formalize the proposition that swept volume measure is Lipschitz continuous and bounded along a continuous trajectory in finite c-space, justifying using DNNs as approxmators.

Proposition 1. *$|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ along a continuous trajectory between two configuration points $\boldsymbol{c}_1, \boldsymbol{c}_2$ in c-space is Lipschitz continuous, i.e.*

$$||\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)| - |\mathcal{SV}(\boldsymbol{c}_3, \boldsymbol{c}_4)|| \leq K\|(\boldsymbol{c}_1, \boldsymbol{c}_2) - (\boldsymbol{c}_3, \boldsymbol{c}_4)\|, \tag{10}$$

where K is a positive constant.

The proof of the proposition is in the Appendix (Section 12.1), and here we provide the intuition behind it. It is known that the swept volume measure of a rigid body undergoing rotation and translation motion is Lipschitz continuous (Schymura 2014). For a non-deformable robot, each rigid body linkage undergoes rotation and translation motion as the robot moves between two configurations. Hence, the swept volume of each linkage is Lipschitz continuous. Since the union of the swept volume of each linkage is smaller than the sum, the swept volume continuity property extends to the whole robot.

6 Evaluation

In this section, we evaluate the deep swept volume measure estimator ($\mathcal{D}^{\text{sv}}_{\text{dnn}}$). Accuracy of the estimator is assessed on seven robots, including a 15 DOF planar manipulator (15 DOF Manipulator), a free-floating rigid body (Rigid Body), a fixed-based Kuka manipulator (Kuka), a robot with closed-loop kinematic chain (Closed-loop), a robot with both revolute and prismatic joints (Prismatic), a Youbot mobile manipulator (Youbot) and a parallel robot (Parallel). Pictures of these robots are in Figure 4. We train a $\mathcal{D}^{\text{sv}}_{\text{we}}$ model and a $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ model to learn $|\widetilde{\mathcal{SV}}_0|$ for each robot. To demonstrate applications for motion planning, we compare PRMs and RRTs that use our estimator (through HNS_{sv}) to ones that use Euclidean $\mathcal{D}_{\text{E}}$ and weighted Euclidean $\mathcal{D}^{\text{sv}}_{\text{we}}$ distance, the most widely used distance measures for sampling-based planners (Palmieri and Arras 2015; Amato et al. 1998).

We highlight important settings used in our evaluation below and provide full details in the Appendix (Section 12.2). The seven DNNs used to learn $|\widetilde{\mathcal{SV}}_0|$ of each robot share the same hyper-parameters and training dataset construction parameters. The dataset has one hundred thousand training samples per robot. We generate additional ten thousand evaluation samples in the same manner, but are unseen by the estimators. In order to satisfy the closed-loop kinematic constraint for the Closed-loop robot, the c-space trajectory between the start and end configurations during training data generation is computed by a DLS-based inverse kinematic technique (Wampler 1986) w.r.t. a straight line end-effector motion. We use PRM and RRT implemented in OMPL (Şucan et al. 2012). PRM with HNS_{sv} identifies $k_{nn} = 5$ nearest neighbors among $k_c = 10$ candidates to connect to, while RRT with HNS_{sv} finds $k_c = 5$ candidates in order to identify the nearest configuration in the tree. Figure 5 shows the start and goal configurations of the three robots used for planning in four environments. To demonstrate one learned model used for multiple planning scenarios, the Kuka manipulator is evaluated in two pick-and-place inspired tasks: `Retrieve` (Figure 5(e, f)) and `Shuffle` (Figure 5(g, h)) with complex environments.

6.1 Learning Results

Table 1 shows the weights of $\mathcal{D}^{\text{sv}}_{\text{we}}$ for various robots. As expected, these weights indicate that revolute joints near the base typically impact $|\widetilde{\mathcal{SV}}_0|$ more. The weights of the Rigid Body and Youbot indicate that the translational DOFs have a higher impact on $|\widetilde{\mathcal{SV}}_0|$ than rotational degrees. Finally, there's no clear pattern for the weights of prismatic joints or joints of the Closed-loop robot. This is likely due to $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ of these two robots being too complex to be described by the simple weighted Euclidean distance model.

15 DOF Manipulator	[214, 97, 80, 73, 60, 43, 31, 28, 23, 19, 8, 5, 9, 7, 5]
Rigid Body	[120, 140, 140] for x, y, z and [86, 110, 82, 88] for quaternions
Kuka	[1506, 2371, 181, 482, 1, 170, 30]
Closed-loop	[262, 1525, 7060, 11892, 542, 84, 3571, 8394, 3161]
Prismatic	[9759, 5284, 3752, 2759] for revolute joints and [5946, 6057, 5908, 5923] for prismatic joints
Youbot	[1718, 1735, 29] for x, y, yaw and [9, 41, 18, 5, 0.9] for manipulator joints
Parallel	[2624, 13252, 59373, 14660, 8948] for leg 1, [6337, 13327, 62128, 8408, 9301] for leg 2, [6431, 13359, 65603, 8431, 9105] for leg 3

Table 1. Weights rounded to integer of weighted Euclidean distance ,$\mathcal{D}^{\text{sv}}_{\text{we}}$, for robots tested.

Figure 6(a)-(g) shows the evaluation L2 loss of $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ and $\mathcal{D}^{\text{sv}}_{\text{we}}$ at each epoch while Figure 6(h) shows the final epoch loss mean and standard deviation. It is clear that five learning instances (each with a different network random initialization) converges for both $\mathcal{D}^{\text{sv}}_{\text{we}}$ and $\mathcal{D}^{\text{sv}}_{\text{dnn}}$, but $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ has a much smaller final loss than $\mathcal{D}^{\text{sv}}_{\text{we}}$ across all robots. This means that $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ learns to approximate $|\widetilde{\mathcal{SV}}_0|$ much better than $\mathcal{D}^{\text{sv}}_{\text{we}}$. In addition, the standard deviation of the loss (represented by the shade) are small for both $\mathcal{D}^{\text{sv}}_{\text{we}}$ and $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ across all robots. This indicates that learning is robust against random network initialization seeds.

Figure 4 shows the histogram of $\mathcal{D}_{\text{E}}$, $\mathcal{D}^{\text{sv}}_{\text{we}}$ and $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ for the evaluation data compared to the ground truth $|\widetilde{\mathcal{SV}}_0|$ (gray shade). Note that in order to compare to $|\widetilde{\mathcal{SV}}_0|$, we scale the value of $\mathcal{D}_{\text{E}}$ such that the average matches the average $|\widetilde{\mathcal{SV}}_0|$ of the evaluation data. We also explored the distance measure performance by comparing $|\widetilde{\mathcal{SV}}_0|$ against each distance measure in a scatter plot (Figure 7). Both Figures

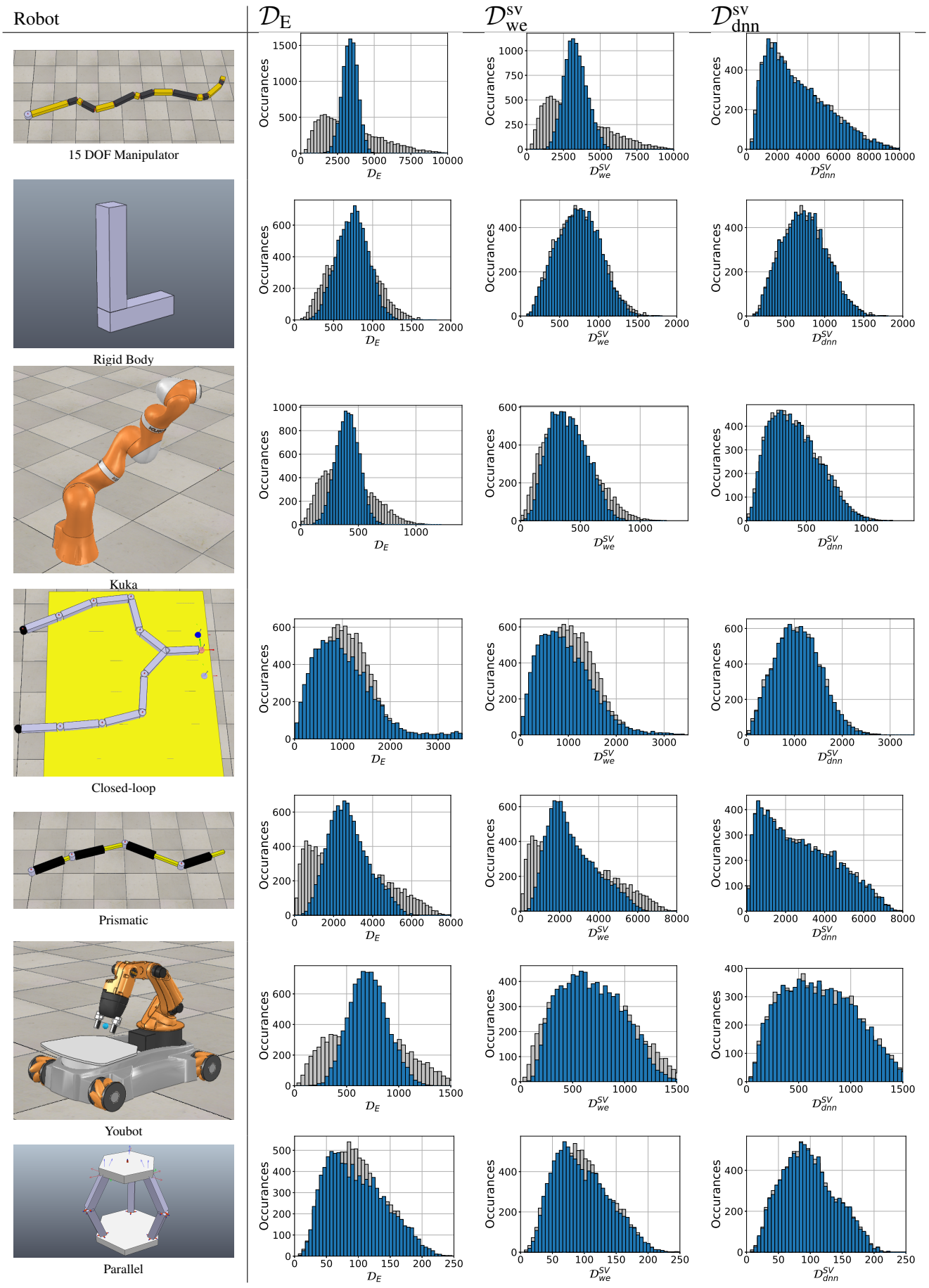


Figure 4. Pictures of robots tested and corresponding histograms of $\mathcal{D}_{\mathrm{E}}$, $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ and $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$. The gray shade is the histogram of ground truth ($|\widetilde{\mathcal{SV}}_0|$). The start and end end-effector positions are sampled in the yellow region for the Closed-loop robot. For the Prismatic robot, the yellow links are connected to black linkages via prismatic joints.

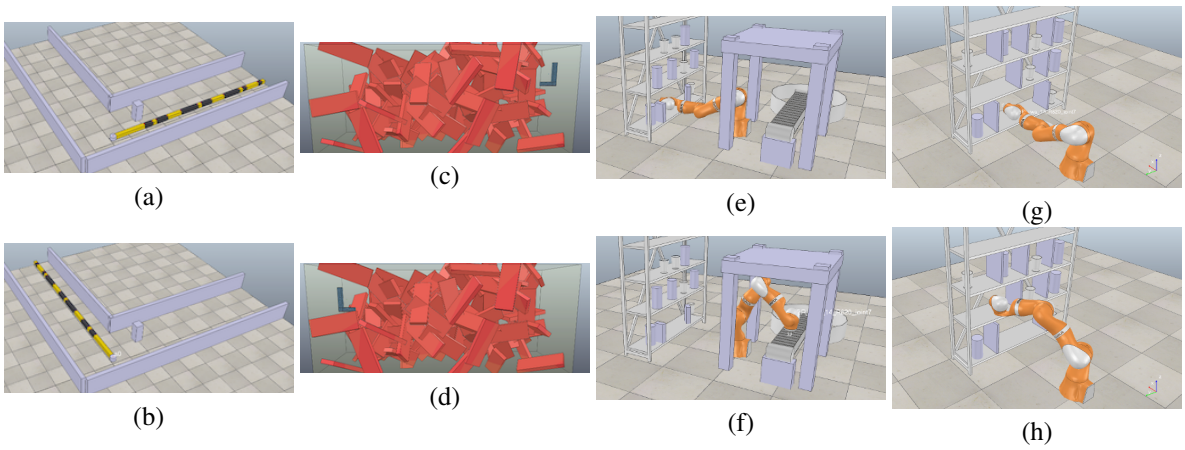
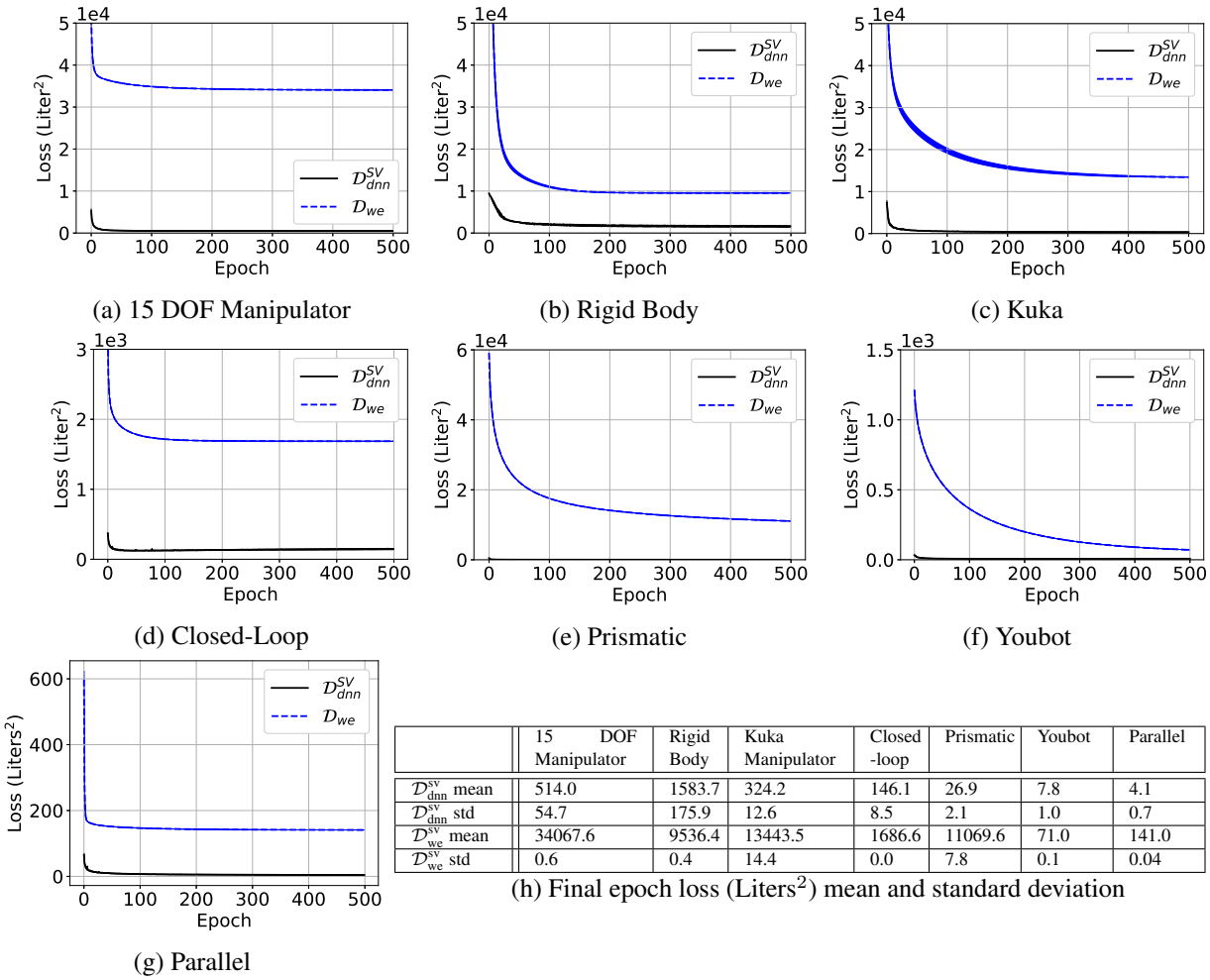


Figure 5. Start (top row) and goal (bottom row) configurations of the 15 DOF planar manipulator (a, b), free-floating Rigid Body (c, d) and Kuka LBR iiwa 14 R820 manipulator in `Retrieve` task(e, f) `Shuffle` task (g, h).



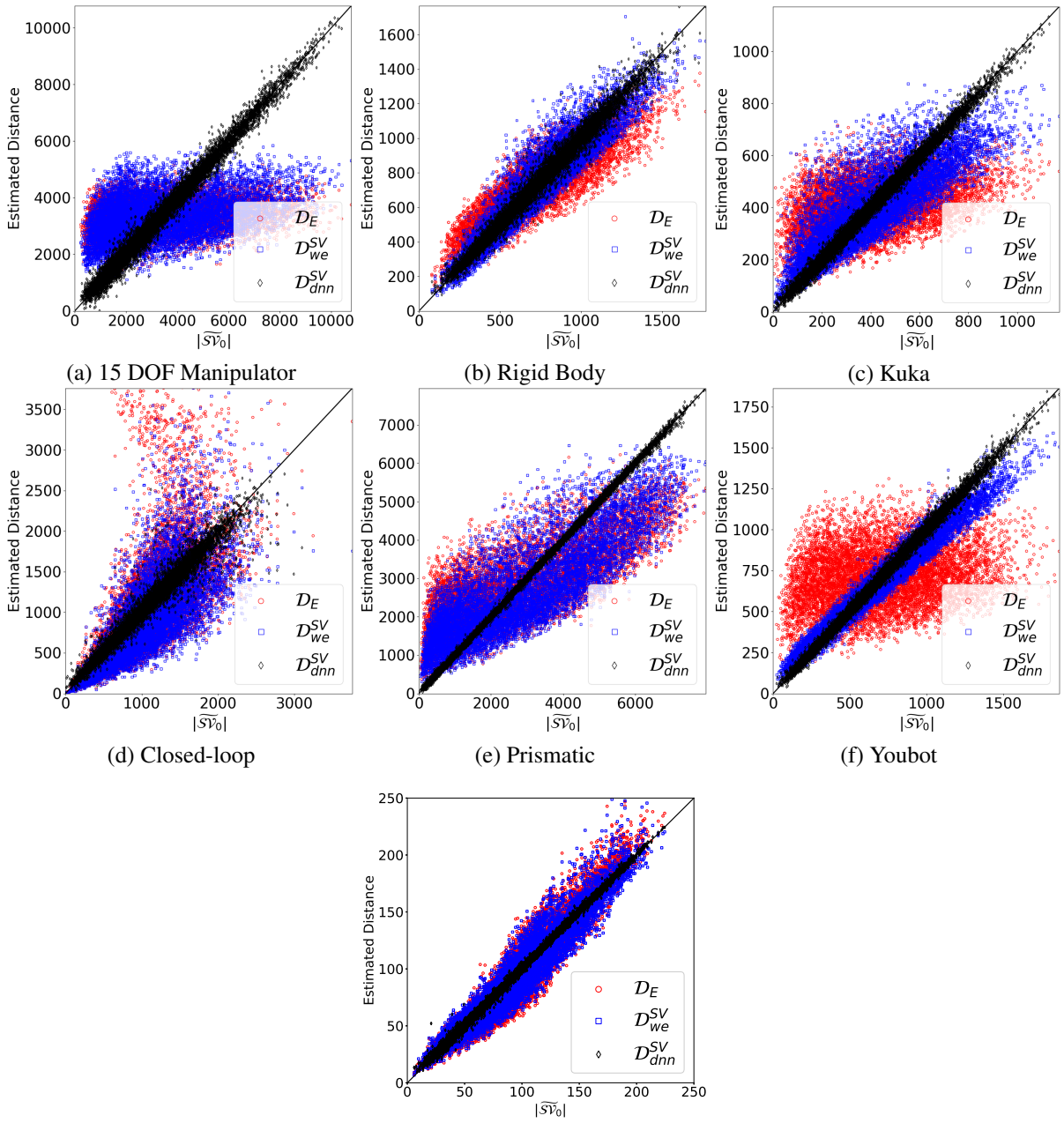
(h) Final epoch loss (Liters^2) mean and standard deviation

	15 DOF Manipulator	Rigid Body	Kuka Manipulator	Closed -loop	Prismatic	Youbot	Parallel
$\mathcal{D}^{sv}_{dnn}$ mean	514.0	1583.7	324.2	146.1	26.9	7.8	4.1
$\mathcal{D}^{sv}_{dnn}$ std	54.7	175.9	12.6	8.5	2.1	1.0	0.7
$\mathcal{D}^{sv}_{we}$ mean	34067.6	9536.4	13443.5	1686.6	11069.6	71.0	141.0
$\mathcal{D}^{sv}_{we}$ std	0.6	0.4	14.4	0.0	7.8	0.1	0.04

Figure 6. Learning curves of $\mathcal{D}^{sv}_{dnn}$ and $\mathcal{D}^{sv}_{we}$. The shade represents the standard deviation of the loss across five training instances each with a different random network initialization.

4 and 7 show that $\mathcal{D}^{sv}_{dnn}$ approximates $|\widetilde{\mathcal{SV}}_0|$ well across all robots. This can be seen by the striking similarities in the $\mathcal{D}^{sv}_{dnn}$ column in Figure 4. In addition, $\mathcal{D}^{sv}_{dnn}$ (black squares in Figure 7) closely tracks $|\widetilde{\mathcal{SV}}_0|$ along the diagonal, indicating good correlation across robot motions with various amounts of $|\widetilde{\mathcal{SV}}_0|$. Note that $\mathcal{D}^{sv}_{dnn}$ performs well even for the robot with closed-loop kinematic chains (Close-loop and Parallel). This is impressive since unlike other robots, the non-linear c-space trajectory between $\boldsymbol{c}_1$, $\boldsymbol{c}_2$ is identified by solving inverse kinematics to satisfy the close-loop constraints.

In contrast to $\mathcal{D}^{sv}_{dnn}$, $\mathcal{D}_E$ (red circles in Figure 7) only correlate to $|\widetilde{\mathcal{SV}}_0|$ well on the Rigid Body. Similarly, $\mathcal{D}^{sv}_{we}$ (blue diamonds in Figure 7) correlates well on only on Rigid Body and Youbot. As demonstrated in Table 1, the $|\widetilde{\mathcal{SV}}_0|$ of these robots are both dominated by the translational DOFs. Note that $\mathcal{D}_E$ and $\mathcal{D}^{sv}_{we}$ are the most commonly used distance



(a) 15 DOF Manipulator (b) Rigid Body (c) Kuka

(d) Closed-loop (e) Prismatic (f) Youbot

(g) Parallel

Figure 7. Scatter plots of $|\widetilde{\mathcal{SV}}_0|$ and the distance estimated by $\mathcal{D}_{\text{E}}$ (red circles), $\mathcal{D}^{\text{sv}}_{\text{we}}$ (blue squares) and the DNN ($\mathcal{D}^{\text{sv}}_{\text{dnn}}$) (black diamonds) for robots tested.

measure for sampling-based motion planning, yet, when the robot has revolute joints, these measures do not correlate well to the volume swept by the robot, which reflects the collision probability between configurations. This is particularly significant for high DOF robots due to revolute joints in articulated bodies impacting each other non-linearly. In addition, the simple $\mathcal{D}^{\text{sv}}_{\text{we}}$ has limited expressive capability. Thus, it is not surprising that it failed to capture the non-linear impact of prismatic joints or the non-linear c-space trajectories imposed by the closed-loop kinematics constraint (Figure 7e and 7d). As a result, the advantage over $\mathcal{D}_{\text{E}}$ on the Closed-loop and Prismatic robots is negligible despite that the weights of $\mathcal{D}^{\text{sv}}_{\text{we}}$ are tuned to approximate $|\widetilde{\mathcal{SV}}_0|$.

Similar trends can be found in Table 2, which shows the L1 norm of the error ratio for various distance metrics and robots w.r.t. the evaluation dataset. The average error ratio of $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ is between 2.2% and 8.1% across all robots tested. In contrast, the average error ratio is 4.38 to 29.75 times larger for $\mathcal{D}_{\text{E}}$ and 2.44 to 22.46 times larger for $\mathcal{D}^{\text{sv}}_{\text{we}}$. We empirically found that the error increases dramatically when DNNs are trained on $\boldsymbol{c}_1 - \boldsymbol{c}_2$ as the input instead of $(\boldsymbol{c}_1, \boldsymbol{c}_2)$. This is expected as $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ is a complex, nonlinear function and a mapping function between $\boldsymbol{c}_1 - \boldsymbol{c}_2$ and $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ may not exist.

We further explored the DNNs' prediction errors of swept volume. The errors are typically very small. Predictions with large errors, e.g., those with more than twice the true volume, occur rarely. For example, less than 1% of the evaluation samples for all robots exhibited this large error: at a minimum, only 0.01% for Rigid Body and at a maximum, 0.23% for Closed-Loop. This can be expected as an artifact

	15 DOF Manipulator	Rigid Body	Kuka Manipulator	Closed -loop	Prismatic	Youbot	Parallel
$\mathcal{D}_{\mathrm{E}}$	76.9%	19.7%	60.7%	66.2%	83.3%	33.0%	11.6%
$\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$	70.7%	11.0%	29.9%	13.3%	62.9%	29.5%	10.6%
$\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$	**8.1%**	**4.5%**	**3.7%**	**3.8%**	**2.8%**	**7.3%**	**2.2%**

Table 2. L1 norm of error ratio $((\mathcal{D} - |\widetilde{\mathcal{SV}}_0|)/|\widetilde{\mathcal{SV}}_0|)$ for the deep swept volume measure estimator ($\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$) and comparison. The measure with the lowest error ratio is highlighted for each robot.

of the employed training data, as these errors typically only occur for very small motions which are typically undersampled. While these prediction errors could be problematic for some fine motion applications, we do note that sampling-based motion planning has been shown to be highly robust to some noise in the neighbor predictions, even potentially enhancing planning quality (McMahon et al. 2012).

6.2 Planning Results

We evaluate the performance impact of the deep swept volume measure estimator (through $\mathrm{HNS}_{\mathrm{sv}}$) on PRM and RRT. The planning environments are shown in Figure 5. The planner performance is shown in Figure 8, where the top row shows the cumulative success rate of identifying a collision-free motion plan as a function of time for PRM and RRT using the $\mathcal{D}_{\mathrm{E}}$ (red), $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ (blue) and $\mathrm{HNS}_{\mathrm{sv}}$ (black) distance measures in various environments. Across all environments and planners, using $\mathrm{HNS}_{\mathrm{sv}}$ is more likely to find a solution within the time budget. The gain in success rate at 200s (max planning time allowed) is between 1.27x to 5x (over $\mathcal{D}_{\mathrm{E}}$) and 1.08x to 2.14x (over $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$). In addition, the bottom row of Figure 8 shows that paths identified by $\mathrm{HNS}_{\mathrm{sv}}$ have a smaller swept volume measure. Comparing across robots, results demonstrate that the advantage of $\mathrm{HNS}_{\mathrm{sv}}$ is much less prominent for the Rigid Body robot. This is expected since $\mathcal{D}_{\mathrm{E}}$ and $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ both approximate $|\widetilde{\mathcal{SV}}_0|$ reasonably well for this robot. $\mathrm{HNS}_{\mathrm{sv}}$ shows a similar performance gain for both the 3D Kuka and the 2D 15 DOF manipulators. In the RRT case, $\mathrm{HNS}_{\mathrm{sv}}$ also enhances planning by identifying solutions with lower swept volume measures and identifying solutions where other distance measures failed. For example, Figure 8a shows that the 15 DOF manipulator in a narrow corridor is very difficult for RRT as neither $\mathcal{D}_{\mathrm{E}}$ nor $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ found a solution in 20 runs. In contrast, RRTs using $\mathrm{HNS}_{\mathrm{sv}}$ were able to identify a solution in 2 runs, likely due to the goal bias mechanism of RRT which can significantly increase the performance of RRT (LaValle and Kuffner 2001). In the planning scenario shown in Figure 5 (a, b), the start and goal have the same joint angles except for the joint at the base. This means $\mathcal{D}_{\mathrm{E}}$ between the start and goal is relatively small. However, the robot must curl towards the base and then extend in order to reach the goal. These curled configurations require a large $\mathcal{D}_{\mathrm{E}}$ change from the goal configuration and therefore are unlikely to be selected by an RRT using goal bias. As a result, the goal bias is ineffective for the Euclidean-based metrics as it mostly selects configurations near the start. In contrast, $\mathrm{HNS}_{\mathrm{sv}}$ does not have this problem since the $|\widetilde{\mathcal{SV}}_0|$ between the start and goal is larger than the $|\widetilde{\mathcal{SV}}_0|$ between any curled configuration and the goal.

7 Physical Robot Experiment

In this section, we demonstrate our method on a physical Baxter robot. The Baxter robot's task is to pick up a red cube in the left compartment and place it in the compartment on the right (Figure 9). Since we only use the right arm for this task, the Baxter robot is modeled as a fixed-based 7 DOF manipulator, where each DOF corresponds to a joint in the right arm.

The correlation between the deep swept volume measure estimator $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ and $|\widetilde{\mathcal{SV}}_0|$ compared to $\mathcal{D}_{\mathrm{E}}$ and $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ is shown in Figure 10. The general trend is very similar to Kuka (Figure 7c), where only $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ approximates $|\widetilde{\mathcal{SV}}_0|$ well. This is not surprising as Kuka and Baxter are both fixed-based 7 DOF manipulators.

The performance of various distance measures with the RRT-Connect planner is shown in Figure 10b and 10c. We chose RRT-Connect since the bi-directional tree-growth provides a significant speed-up in the tested environment, where the start and goal end-effector positions are inside narrow compartments. The planning performance is also similar to Kuka in two ways. First, using $\mathrm{HNS}_{\mathrm{sv}}$ is more likely to find a solution within the time budget than $\mathcal{D}_{\mathrm{E}}$ or $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ (Figure 10b). Next, paths identified by $\mathrm{HNS}_{\mathrm{sv}}$ have on average, 1.5x smaller (67%) swept volume measure than $\mathcal{D}_{\mathrm{E}}$ and 1.3x (77%) smaller than $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ (Figure 10c). A run of the path execution and the difference in path swept volume measure between $\mathcal{D}_{\mathrm{E}}$ and $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ is included in the enclosed video.

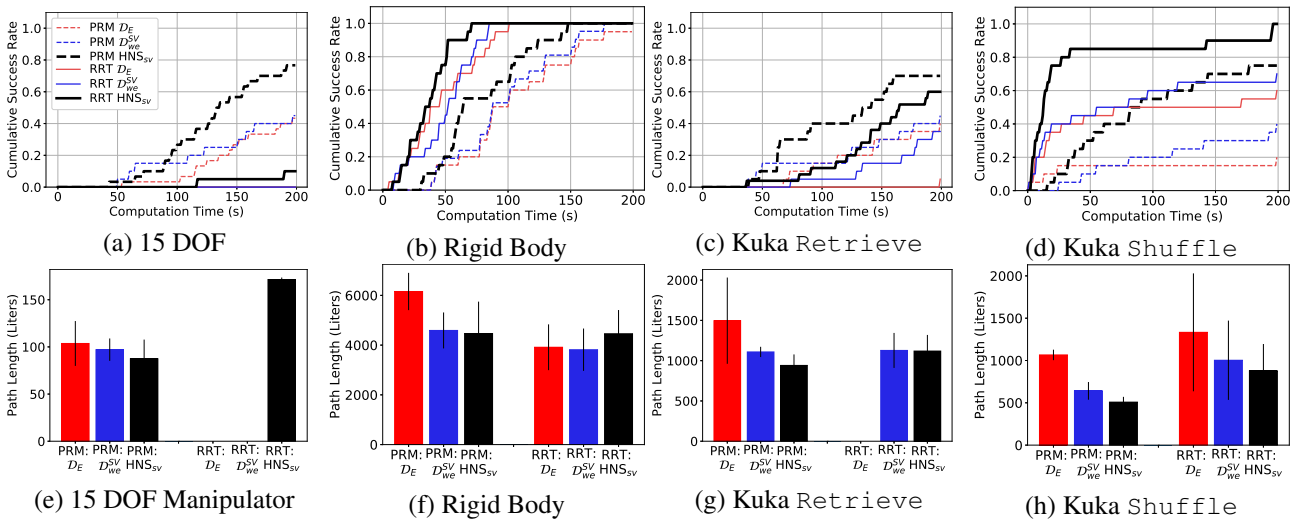
8 Discussion

This section further investigates the advantage and trade-offs of the deep swept volume measure estimator compared to other distance measures (Section 8.1). We also investigate aspects of the estimator that may impact planning performance, such as the DNN and training data size (Section 8.2), and the estimator's ability to generalize to planning environments larger than the one it is trained on (Section 8.3).

8.1 Distance Measure Trade-Offs

In Section 6, the advantages of $\mathrm{HNS}_{\mathrm{sv}}$ are clear, particularly when $\mathcal{D}_{\mathrm{E}}$ or $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ cannot capture $|\widetilde{\mathcal{SV}}_0|$ well, i.e., when the robot has a highly articulated body. Here we investigate the advantages further by empirically evaluating the computational cost and the quality of returned nearest neighbors for each distance measure.

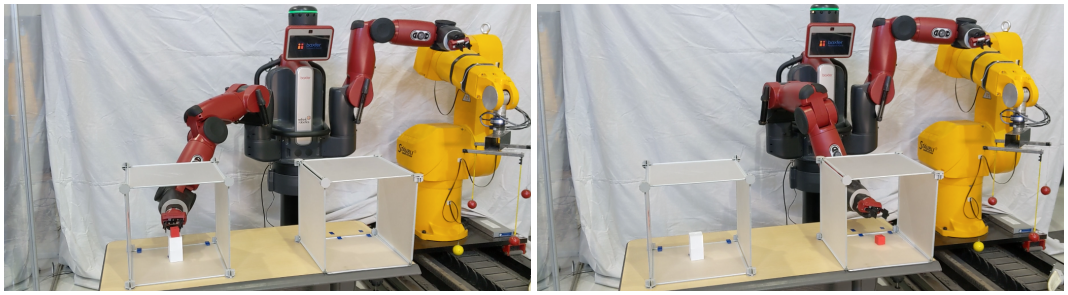
The computation time required by distance measures is shown in Table 3. Recall that $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$ and $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ are trained once per robot (columns 3 and 4) and utilized one hundred thousand training samples (column 2). After training, the computation time for a single inference to $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ is at or just



(a) 15 DOF (b) Rigid Body (c) Kuka `Retrieve` (d) Kuka `Shuffle`

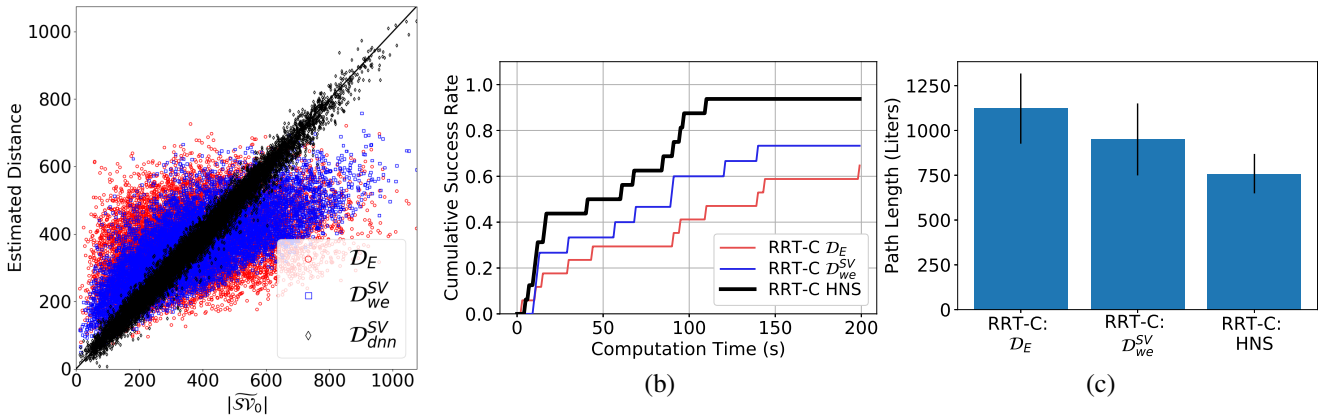
(e) 15 DOF Manipulator (f) Rigid Body (g) Kuka `Retrieve` (h) Kuka `Shuffle`

Figure 8. The cumulative success rate of identifying a path (top row) and path length (in units of $|\widetilde{\mathcal{SV}}|$) of successful runs (bottom row) for PRM (dotted lines) and RRT (solid lines). The color of bars and curves represents various distance measures (red: $\mathcal{D}_{\mathsf{E}}$, blue: $\mathcal{D}_{\mathsf{we}}^{\mathsf{sv}}$, black: $\mathcal{D}_{\mathsf{dnn}}^{\mathsf{sv}}$ (our method)). The path length data is not available for RRTs using $\mathcal{D}_{\mathsf{E}}$ or $\mathcal{D}_{\mathsf{WE}}$ for the 15 DOF Manipulator since there were zero successful runs.



(a) Start (b) Goal

Figure 9. Start and goal configurations of the physical Baxter robot.



(a) (b) (c)

Figure 10. (a) Scatter plot of $|\widetilde{\mathcal{SV}}_0|$ and the distance estimated by $\mathcal{D}_{\mathsf{E}}$ (red circles), $\mathcal{D}_{\mathsf{we}}^{\mathsf{sv}}$ (blue squares) and the DNN ($\mathcal{D}_{\mathsf{dnn}}^{\mathsf{sv}}$) (black diamonds) for Baxter. (b) The cumulative success rate of identifying a path and (c) path length (in units of $|\widetilde{\mathcal{SV}}|$) of successful runs for RRT-Connect.

	Training			Distance Measure Call		
Robot	Data Generation	$\mathcal{D}_{\mathsf{we}}^{\mathsf{sv}}$ Training	$\mathcal{D}_{\mathsf{dnn}}^{\mathsf{sv}}$ Training	Compute $\mathcal{D}_{\mathsf{we}}^{\mathsf{sv}}$	Compute $\mathcal{D}_{\mathsf{dnn}}^{\mathsf{sv}}$	Compute $\widetilde{\mathcal{SV}}$
15 DOF Manipulator	31hr	630.02s	4360.03s	0.081μs	175.1μs	8.85s
Rigid Body	2hr	601.53s	4001.53s	0.053μs	164.3μs	0.58s
Kuka Manipulator	14hr	629.33s	4023.35s	0.055μs	164.3μs	4.06s

Table 3. Computation time of various operations broken down by training (Training) and a single usage as done as a primitive operation in motion planning (Distance Measure Call).

under 175μs (column 6). Comparing a single inference to time required to generate $|\widetilde{\mathcal{SV}}_0|$ for each robot shows that $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ inference is 3500 to 5000 times faster than state of the art $|\widetilde{\mathcal{SV}}|$ computation. In addition, the computation time of $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ is only slightly affected by the DOF of the robot and is independent of the robot's 3D model complexity. On the other hand, computing $\mathcal{D}^{\text{sv}}_{\text{we}}$ (column 5) is about 2000 to 3000 times faster than querying $\mathcal{D}^{\text{sv}}_{\text{dnn}}$. These results suggest HNS_{sv} can reduce computation time as it identifies candidate nearest neighbors using the fast $\mathcal{D}^{\text{sv}}_{\text{we}}$ and efficient data-structures before the slower, more accurate, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$.

From Section 6, it is clear that nearest neighboring configurations selected w.r.t. $\mathcal{D}_{\text{E}}$ and $\mathcal{D}^{\text{sv}}_{\text{we}}$ are very different from ones selected by HNS_{sv}. However, little is known about the quality of neighboring configurations returned by the distance measures. We evaluate this by comparing neighbors returned by each measure to those returned by $|\widetilde{\mathcal{SV}}_0|$. Since a full comparison during a planning run would be computationally prohibitive, we randomly sample 100 starting configurations ($\boldsymbol{c}_1$) and 100 potential neighbor configurations ($\boldsymbol{c}_2$) for each robot. For each $\boldsymbol{c}_1$, five nearest configurations among $\boldsymbol{c}_2$ are identified using $\mathcal{D}_{\text{E}}$, $\mathcal{D}^{\text{sv}}_{\text{we}}$, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ and HNS_{sv} ($k_c = 10$). Then, these configurations are compared to the configurations selected by $|\widetilde{\mathcal{SV}}_0|$. Table 4 captures the quantity and quality differences in the returned neighbor configurations. First, the percentage of configurations returned by each measure that do not match those returned by $|\widetilde{\mathcal{SV}}_0|$ are shown. Next, the quality of the returned configurations for each measure is demonstrated by tallying the additional volume swept by the returned neighbors over the baseline provided by the configurations returned by $|\widetilde{\mathcal{SV}}_0|$. These values demonstrate that $\mathcal{D}_{\text{E}}$ selects very different neighboring configurations than $|\widetilde{\mathcal{SV}}_0|$, in one example incurring a 171% increase in swept volume. In contrast, $\mathcal{D}^{\text{sv}}_{\text{we}}$, with weights optimized to mimic $|\widetilde{\mathcal{SV}}_0|$, chooses neighboring configurations with up to 7% increase in additional volume swept for the L-shaped and Kuka manipulator robot. However, the simple weights face difficulty capturing the highly nonlinear $|\widetilde{\mathcal{SV}}_0|$ of the 15 DOF manipulator well, resulting in a 166% increase in volume swept. In contrast, HNS_{sv} chooses neighboring configurations closest to $|\widetilde{\mathcal{SV}}_0|$, and the additional volume swept is much lower than any other Euclidean-based measure, i.e., 2.7x (37%) to 3.5x (28%) smaller than $\mathcal{D}^{\text{sv}}_{\text{we}}$. As expected, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ selects neighboring configurations closest to $|\widetilde{\mathcal{SV}}_0|$ for all robots tested. However, computing $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ is much slower than HNS_{sv}, and efficient nearest neighbor data structures cannot be used since $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ does not form a metric space.

To further investigate the trade-offs between HNS_{sv} and $\mathcal{D}^{\text{sv}}_{\text{dnn}}$, we compare motion planners using $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ directly as a distance measure against HNS_{sv} and other baselines. The magenta curves and bars in Figure 11a and 11b show the performance of $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ for the Kuka robot in the `Shuffle` task. In the RRT case (solid lines), $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ outperforms $\mathcal{D}_{\text{E}}$ and $\mathcal{D}^{\text{sv}}_{\text{we}}$ and performs similarly to HNS_{sv}. On the other hand, in the PRM case, $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ performs similarly to $\mathcal{D}_{\text{E}}$ and performs much worse than HNS_{sv}. This is likely due to PRM making more $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ queries (about 450,000 on average) than RRT (about 150,000) in this planning scenario. Therefore, despite the selection of neighbors with small $|\widetilde{\mathcal{SV}}_0|$ by $\mathcal{D}^{\text{sv}}_{\text{dnn}}$, the longer computation time negatively impacts planner performance under a fixed time budget. Paths identified by $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ generally have smaller swept volume measure compared to HNS_{sv} and baselines, especially in the PRM case. This is expected as planners using $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ selects neighbors with smaller $|\widetilde{\mathcal{SV}}_0|$ to connect.

Overall, HNS_{sv} offers the best balance between path quality and planning efficiency compared to $\mathcal{D}_{\text{E}}$, $\mathcal{D}^{\text{sv}}_{\text{we}}$ and $\mathcal{D}^{\text{sv}}_{\text{dnn}}$. HNS_{sv} achieves this through the hierarchical combination of fast $\mathcal{D}^{\text{sv}}_{\text{we}}$, efficient nearest neighbor data-structure and accurate $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ queries.

8.2 Network and Training Data Size

We also investigate the learning performance of DNNs as impacted by the number of neurons in the hidden layer and the quantity of training samples. Figure 12 shows the L2 loss over the evaluation dataset as a function of training epochs as impacted by combinations of training sample and DNN sizes. It is clear that networks trained with 25,000 training samples (1/4 the original quantity of samples, shown as dashed lines) have higher loss across all robots and exhibit over-fitting as the loss increases after the initial decrease. When trained with the full training dataset (solid curves), large networks (networks with 1024 and 512 neurons in the first hidden layer) perform similarly across all robots while small networks demonstrate a larger loss for Kuka and Rigid Body robots. These results indicate that a large network and training dataset size are important to accurately approximate $|\widetilde{\mathcal{SV}}_0|$.

8.3 Generalization of Translational DOFs

Recall that the deep swept volume measure estimator is trained by $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ samples generated in a finite-sized environment, yet, motion planning often occurs in large environments. As a result, the translational DOFs of the start and end configurations queried during motion planning may differ significantly from ones the network was trained on. We investigate this in Figure 13 by evaluating the network in environments that are 2X (blue), and 4X (cyan) larger in all translational dimensions than the environment it was trained on (red). We only evaluate Rigid Body and Youbot since other robots do not have translational DOF. For both robots, the DNN generalizes well to the 2X environment, as seen by the clustering along the diagonal line. The accuracy of the estimator is reduced slightly as the spread around the diagonal is larger. For the 4X case, the DNN still performs well but the accuracy is slightly reduced further. This is expected as DNNs are function approximators designed to achieve statistical generalization among the training dataset (Goodfellow et al. 2016). Since $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ generalize well, we expect the impact on planner performance to be small when using $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ (through HNS_{sv}) in large environments. This ability to generalize is also reflected by the superior performance of HNS_{sv} in the Rigid Body planning experiment, where the environment is 11x bigger than the training environment (Figure 5c and 8d).

	Percent of Non-Matching Neighbors				Percent Additional Volume Swept			
Robot	$\mathcal{D}_{\mathrm{E}}$	$\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$	$\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$	$\mathrm{HNS}_{\mathrm{sv}}$	$\mathcal{D}_{\mathrm{E}}$	$\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$	$\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$	$\mathrm{HNS}_{\mathrm{sv}}$
15 DOF	87%	80%	**32%**	65%	180%	160%	**14%**	61%
Rigid Body	65%	22%	**11%**	**11%**	38%	7%	**2%**	**2%**
Kuka	87%	33%	**12%**	15%	56%	8%	**1%**	2%

Table 4. Comparison of neighboring configurations selected by various distance measures as compared to those selected by $|\widetilde{\mathcal{SV}}_0|$. The Percent of Non-Matching Neighbors columns demonstrate the quantity of neighboring configurations that do not match those selected by $|\widetilde{\mathcal{SV}}_0|$. The Additional Swept Volume columns capture the amount of additional volume swept measure by the neighboring configurations selected by the measures as over that of the $|\widetilde{\mathcal{SV}}_0|$ configurations. The best measure of each robot is highlighted.

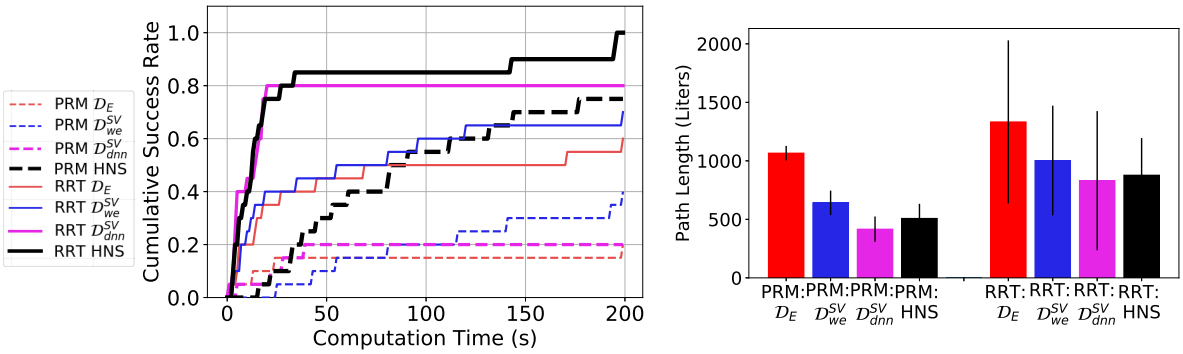
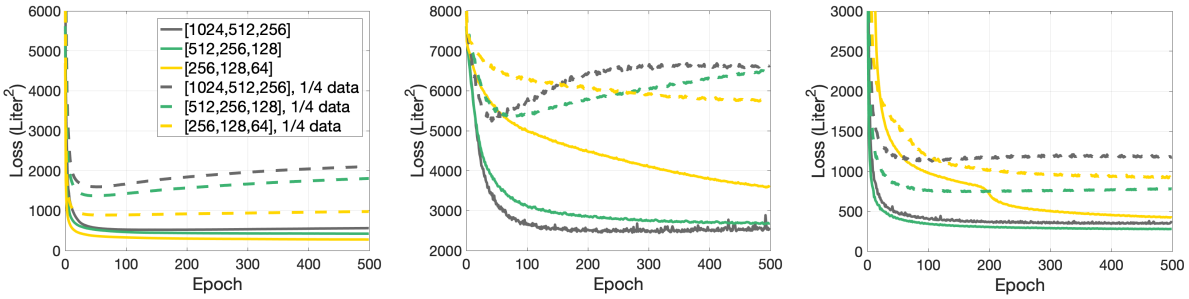


Figure 11. (a) Cumulative success rate of identifying a path for PRM (dotted lines) and RRT (solid lines). (b) Path length (in units of $|\widetilde{\mathcal{SV}}_0|$) of successful runs evaluated on the Kuka manipulator in the `Shuffle` task. The color of bars and curves represents various distance measure (red: $\mathcal{D}_{\mathrm{E}}$, blue: $\mathcal{D}^{\mathrm{sv}}_{\mathrm{we}}$, magenta: $\mathcal{D}^{\mathrm{sv}}_{\mathrm{dnn}}$ and black: $\mathrm{HNS}_{\mathrm{sv}}$).



(a) 15 DOF Manipulator (b) Rigid Body (c) Kuka

Figure 12. The L2 evaluation loss of DNNs with varied numbers of neurons in the hidden layer as shown in the legend (solid curves) across the (a) 15 DOF manipulator, (b) Rigid Body and (c) Kuka manipulator. The dashed curves show the same network trained with 25,000 training samples (1/4 of the full sample size).

9 Deep Swept Volume Geometry Estimator

In this section, we demonstrate as a proof of concept that DNNs can also estimate the swept volume geometry ($\widetilde{\mathcal{SV}}$) accurately. This enables applications such as bridging between task and motion planning (Gaschler et al. 2013; Kaelbling and Lozano-Pérez 2010). We highlight the important setup details here and leave the specifics in the Appendix (Section 12.3).

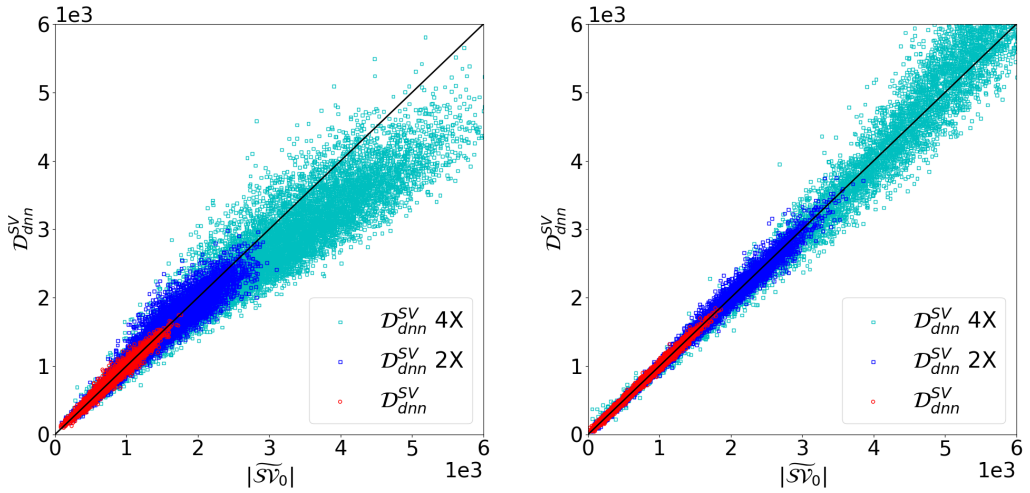
The training data is generated in the same method as described in Section 4.2 except for one modification. Instead of using $|\widetilde{\mathcal{SV}}_0(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ as the ground truth label, we use $\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$ (Eq. 5) approximated by a uniform workspace occupation grid (voxels) with resolution Δ'. The deep swept volume geometry estimator is also a deep neural network with feed-forward fully-connected neurons. The input of the network is the start and end configurations and the output is the probability of occupation for each voxel. The network training is also the same, with the exception of the loss is replaced by the L_2 loss of all N_{voxels} voxels across n training samples

$$\mathcal{L}_{\mathrm{geometry}} = \sum_{i=1}^{n} \sum_{j=1}^{N_{\mathrm{voxels}}} \left(\widetilde{\mathcal{SV}}'_{\mathrm{dnn},j}(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i}) - \widetilde{\mathcal{SV}}_j(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i}) \right)^2, \tag{11}$$

where $\widetilde{\mathcal{SV}}'_{\mathrm{dnn},j}(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i})$ is the network output for the voxel j, and N_{voxels} is the number of voxels used to approximate $\widetilde{\mathcal{SV}}$. Lastly, to estimate the swept volume geometry, we convert the network output of each voxel to boolean by a threshold $0 < \Theta < 1$

$$\widetilde{\mathcal{SV}}_{\mathrm{dnn},j}(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i}) = \begin{cases} 1, & \widetilde{\mathcal{SV}}'_{\mathrm{dnn},j}(\boldsymbol{c}_{1,i}, \boldsymbol{c}_{2,i}) > \Theta \\ 0, & \text{otherwise.} \end{cases}, \tag{12}$$

We evaluate the deep swept volume geometry estimator ($\widetilde{\mathcal{SV}}_{\mathrm{dnn}}$) on the Kuka robot. Figure 14 shows an example of $\widetilde{\mathcal{SV}}$ (ground truth, right), $\widetilde{\mathcal{SV}}_{\mathrm{dnn}}$ (estimated, middle) and the difference (left). The striking similarity between $\widetilde{\mathcal{SV}}$ and $\widetilde{\mathcal{SV}}_{\mathrm{dnn}}$ indicates that swept volume geometries can be learned



(a) Rigid Body $\mathcal{D}^{\text{sv}}_{\text{dnn}}$ (b) Youbot $\mathcal{D}^{\text{sv}}_{\text{dnn}}$

Figure 13. Scatter plots of $|\widetilde{\mathcal{SV}}_0|$ and the distance estimated by DNN ($\mathcal{D}^{\text{sv}}_{\text{dnn}}$). The DNNs are trained in a small environment and evaluated in environments that are the same (red), 2x (blue), and 4x (cyan) larger in all translational dimensions.

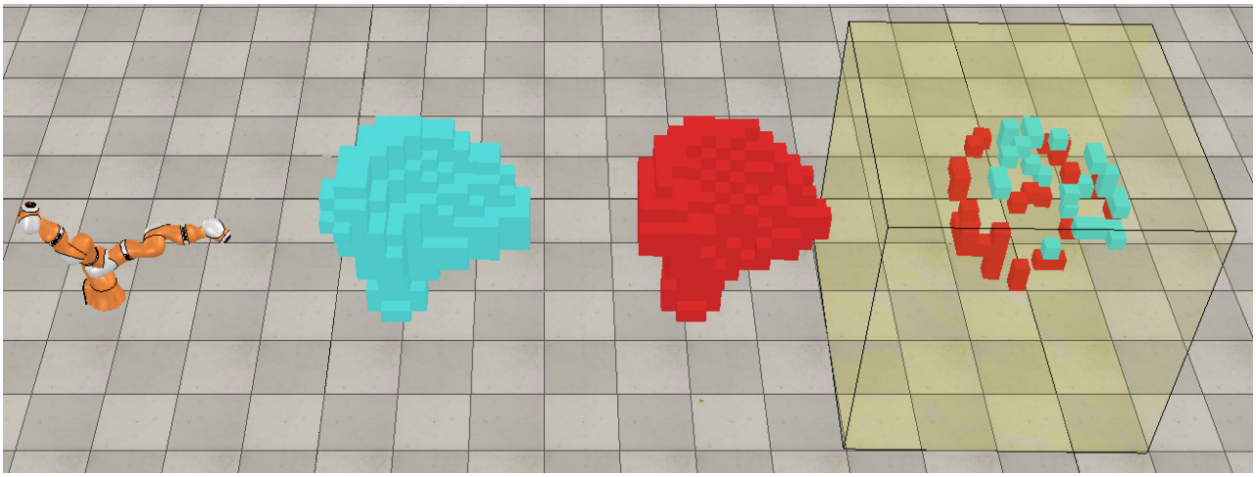


Figure 14. Example start and end configurations of Kuka (left) and the resulting ground truth swept volume geometry $\widetilde{\mathcal{SV}}$ (middle, blue), the estimated swept volume geometry $\widetilde{\mathcal{SV}}_{\text{dnn}}$ (middle, red) and the difference (right, false negatives are in blue and false positives in red). The yellow box shows the extent of the 20x20x20 voxel grid. 60 voxels were estimated erroneously and while 7940 voxels were estimated correctly.

accurately by DNN. Erroneous predictions (false positive and false negatives) occur mostly near the surface of $\widetilde{\mathcal{SV}}$. This is further supported by Figure 15, which shows the Euclidean distance of erroneously predicted voxels to the surface of $\widetilde{\mathcal{SV}}$. Over 99% of erroneously predicted voxels have an Euclidean distance less than or equal to $\sqrt{2}$ voxels. This indicates that most prediction errors are adjacent to the surface of $\widetilde{\mathcal{SV}}$. This is a desired property when the estimator is used to bridge between task and motion planning since the shape of the estimated swept volume remains similar to $\widetilde{\mathcal{SV}}$.

Next we quantify the estimator accuracy in Table 5, which shows the confusion matrix, accuracy, precision and recall of the estimator with $\Theta = 0.4$ over the evaluation dataset. The estimator has a very high accuracy (98.8%) and correctly estimated that more than 95% of the voxels are unoccupied on average. The estimator also has a low false positive and false negative rate, as captured by the high 86.3% recall and 87.1% precision. Note that statistics in Table 5 are functions

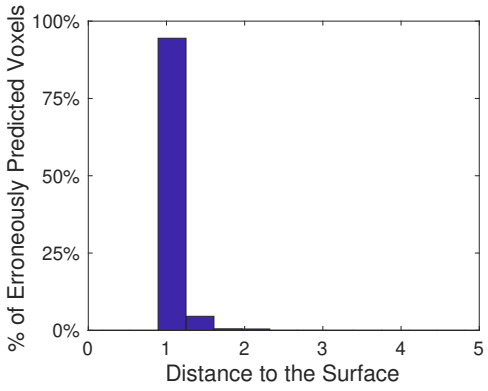


Figure 15. Histogram of Euclidean distance to the surface of $\widetilde{\mathcal{SV}}$ for all voxels erroneously estimated by $\widetilde{\mathcal{SV}}_{\text{dnn}}$ (false positive and false negative). The distance unit is the voxel width (0.1m).

	Prediction On	Prediction Off
Ground Truth On	3.84%	0.57%
Ground Truth Off	0.66%	94.98%
Accuracy: 0.988	Precision: 0.872	Recall: 0.863

Table 5. Confusion matrix, accuracy, precision and recall of the deep swept volume geometry estimator ($\widetilde{\mathcal{SV}}_{\text{dnn}}$) with the threshold value of 0.4. "On" represents the swept volume occupies a voxel.

of the threshold Θ given a fixed network $\widetilde{\mathcal{SV}}'_{\text{dnn}}$. We chose $\Theta = 0.4$ that maximizes the F1 score ($2\frac{Precision \times Recall}{Precision + Recall}$) of 0.8675 in order to balance between precision and recall.

$\widetilde{\mathcal{SV}}_{\text{dnn}}$ Batch 100	$\widetilde{\mathcal{SV}}_{\text{dnn}}$ Batch 10	$\widetilde{\mathcal{SV}}_{\text{dnn}}$ Batch 1	$\widetilde{\mathcal{SV}}$
0.72±0.04	3.47±0.14	9.72±0.75	891.21±95.44

Table 6. Computation time (ms) of estimating $\widetilde{\mathcal{SV}}$ represented by a 20x20x20 voxel grid for various batch sizes (number of $\widetilde{\mathcal{SV}}$ estimates per DNN inference) compared to an octree-based method ($\widetilde{\mathcal{SV}}$). The units are in ms.

We investigate the computation time of the estimator in various batch sizes, i.e., number of $\widetilde{\mathcal{SV}}$ estimates per network inference (Table 6). It is clear that our estimator is much faster than computing $\widetilde{\mathcal{SV}}$ using an octree-based swept volume algorithm. Specifically, the batch 1 column shows that the computation time to make one $\widetilde{\mathcal{SV}}$ estimate is 92x faster than the octree-based method. Furthermore, when swept volume geometry calls can be batched together, the overhead of initializing the network is further reduced, making it over 250x to 1200x faster (batch 10 and batch 100 columns).

10 Conclusion

The ability of DNNs to approximate any continuous bounded function makes them especially well suited to estimate swept volumes. We demonstrated this ability comprehensively on systems such as Rigid Body, closed-loop kinematic chains and manipulators with rotation, prismatic or translation motions. We developed techniques and demonstrate that DNNs estimating the size of swept volume can significantly speed up sampling-based motion planning and improve solution quality in both physical and simulated environments. This advantage was investigated further by exploring the trade-offs between distance measures. Lastly, as a proof of concept, we also demonstrate that the geometry of swept volumes can also be accurately predicted by DNNs. In future work, we plan to use the deep swept volume geometry estimator as a bridge between task and motion planning.

11 Acknowledgement

The authors thank Dr. Andrew Ferdinand, Shakeeb Ahmed and Prof. Rafael Fierro for their help with physical robot experiments. Tapia, Chiang, and Sugaya are partially supported by the National Science Foundation under Grant Numbers IIS-1528047 and IIS-1553266. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation. Tapia, Sugaya, and Yousefi are partially supported by the Air Force Research Laboratory (AFRL) under agreement number FA9453-18-2-0022. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

12 Appendix

12.1 Proof of Proposition 1

Consider the c-space of a non-deformable robot $\mathcal{R}$ with d_f DOF, operating in 3-dimensional workspace. The robot consists of n-joints connecting total of $n+1$ rigid body linkage. Motion of each joint $i = 1, \ldots, n$ can rotate or translate all the successive linkage, i.e., $i+1, \cdots, n+1$ linkage, as a rigid body. For example, in the case of rotation with respect to i^{th} link, a spherical joint rotates the linkage along up to the three rotational DoF, yaw, pitch and roll, denoted by ρ_i, θ_i, ψ_i, respectively. In another example, in the case of translation, a prismatic joint translates the linkage along the axis of i^{th} link, u_i. Each joint motion consists of a single or combination of up to these four DOF motions. Under these assumptions a configuration point in the c-space can be a $d_f = 6 + 4n$ dimensional vector, $\boldsymbol{c} = (x, y, z, \rho, \theta, \psi, u_1, \rho_1, \theta_1, \psi_1, \ \cdots, u_n, \rho_n, \theta_n, \psi_n)$, where the first six DOF are the Cartesian coordinates, x, y, z, and the rotational DOF, ρ, θ, ψ, of the first link (the base of a robot) in the global coordinate frame. This c-space is general and can fully describe free-floating rigid and articulated bodies. Keeping the notation consistent with the main text, let us denote the j^{th} configuration in c-space of a n-jointed robot $\mathcal{R}$ by

$$\boldsymbol{c}_j^n = (x^{(j)}, y^{(j)}, z^{(j)}, \rho^{(j)}, \theta^{(j)}, \psi^{(j)}, u_1^{(j)}, \rho_1^{(j)}, \theta_1^{(j)}, \psi_1^{(j)}, \ldots, u_n^{(j)}, \rho_n^{(j)}, \theta_n^{(j)}, \psi_n^{(j)}), \quad (13)$$

where we may write $\boldsymbol{c}_j = \boldsymbol{c}_j^n$ for short.

Proof. To show that the Lipschitz continuity extends to the cases where the robot changes shape due to i^{th} joint rotations or translation, we use mathematical induction by the linkage.

(Base Case: $i = 1$) $|\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)|$ of a rigid body with respect to translation and rotation is Lipschitz continuous per Schymura (2014).

(IH Case: $i = 2, \ldots, n-1$) Assume that the size of swept volume for the linkage is Lipschitz continuous w.r.t. the rotation and translation of all the joints.

(Step Case: $i = n$) We need to show that (10) holds, for $(\boldsymbol{c}_1, \boldsymbol{c}_2), (\boldsymbol{c}_3, \boldsymbol{c}_4) \in \mathbb{R}^{(6+4n)} \otimes \mathbb{R}^{(6+4n)}$. To do this, let us consider the motions of the first $i = i+1, \ldots, n$ linkage and the last $i = n+1$ link separately. Denote the start ($j = 1, 3$) and goal ($j = 2, 4$) configurations for the former by

$$\boldsymbol{c}_j^{n-1} = (x^{(j)}, y^{(j)}, z^{(j)}, \rho^{(j)}, \theta^{(j)}, \psi^{(j)} u_1^{(j)}, \rho_1^{(j)}, \theta_1^{(j)}, \psi_1^{(j)}, \ldots, u_{n-1}^{(j)}, \rho_{n-1}^{(j)}, \theta_{n-1}^{(j)}, \psi_{n-1}^{(j)}), \quad (14)$$

and those of the latter by

$$\boldsymbol{c}^0_{j=\{1,3\}} = (x_n^{(j)}, y_n^{(j)}, z_n^{(j)}, u_n^{(j)}, \rho_n^{(j)}, \theta_n^{(j)}, \psi_n^{(j)}), \quad (15)$$

$$\boldsymbol{c}^0_{j=\{2,4\}} = \boldsymbol{c}^0_{j=\{1,3\}} + \Delta \boldsymbol{c}^0_{j=\{1,3\}}, \quad (16)$$

where
$\Delta \boldsymbol{c}_j^0 = (\Delta x_n^{(j)}, \Delta y_n^{(j)}, \Delta z_n^{(j)}, \Delta u_n^{(j)} \Delta \rho_n^{(j)}, \Delta \theta_n^{(j)}, \Delta \psi_n^{(j)})$.
The coordinate variables with subscript n, x_n, y_n, z_n, are the coordinates of the end of the n^{th} linkage in the global coordinate frame. Let $\boldsymbol{c}_3 = \boldsymbol{c}_1 + \Delta \boldsymbol{c}_1$, $\boldsymbol{c}_4 = \boldsymbol{c}_2 + \Delta \boldsymbol{c}_2$. Then, we can decompose the $|\mathcal{SV}|$ into (since the union is smaller than equal to the sum)

$$
\begin{aligned}
&\| |\mathcal{SV}(\boldsymbol{c}_1^n, \boldsymbol{c}_2^n)| - |\mathcal{SV}(\boldsymbol{c}_3^n, \boldsymbol{c}_4^n)| \| && (17)\\
&\quad \leq \| |\mathcal{SV}(\boldsymbol{c}_1^{n-1}, \boldsymbol{c}_2^{n-1})| - |\mathcal{SV}(\boldsymbol{c}_3^{n-1}, \boldsymbol{c}_4^{n-1})| \| \\
&\qquad + \| |\mathcal{SV}(\boldsymbol{c}_1^0, \boldsymbol{c}_2^0)| - |\mathcal{SV}(\boldsymbol{c}_3^0, \boldsymbol{c}_4^0)| \| && (18)\\
&\quad \leq K_{n-1} \| \Delta \boldsymbol{c}_1^{n-1} + \Delta \boldsymbol{c}_2^{n-1} \| + K_n \| \Delta \boldsymbol{c}_1^0 + \Delta \boldsymbol{c}_2^0 \| && (19)\\
&\quad \leq K \| \Delta \boldsymbol{c}_1^{n-1} + \Delta \boldsymbol{c}_2^{n-1} + \Delta \boldsymbol{c}_1^0 + \Delta \boldsymbol{c}_2^0 \| && (20)\\
&\quad \leq K \| \Delta \boldsymbol{c}_1^n + \Delta \boldsymbol{c}_2^n \|, && (21)
\end{aligned}
$$

where K_{n-1} and K_n are positive real constants, and $K = max(K_{n-1}, K_n)$. The first term of equation (18) satisfies (IH) and the second one satisfies conditions of Schymura (2014). The translation factor (affecting x, y, z) in the second term is the result of the rotational/translational motion of the n^{th} linkage. □

12.2 Robot Details

The 15 DOF planar manipulator has a fixed round base and 15 rigid cuboid bodies connected by 15 joints. The length of the bodies are [0.8, 0.2, 0.1, 0.3, 0.4, 0.5, 0.1, 0.3, 0.4, 0.5, 0.1, 0.1, 0.4, 0.1, 0.1]m long and 0.1m wide while the corridors are 1.5m wide. The square obstacle in the corridor increases the planning difficulty and is 0.1m in width. The 15 joint angles describe a configuration of the robot. Training sample configurations are uniform-randomly sampled from [-π, π] for the base joint and [-$\pi/2$, $\pi/2$] for all other joints.

The L-shaped free-floating rigid body is sized at 0.4m, 0.6m, 0.1m (width, height, depth). During training, the translation DOFs are limited to a 1.5m box. There are also 100 randomly placed rectangular obstacles of size 0.4m, 1.1m, 0.1m in an environment of size 6m, 2.5m, 2.5m. Training sample configurations are uniform-randomly sampled from [-1.5, 1.5]m for the position axes. The planning environment is 11x larger than the learning environment. To ensure uniform sampling for rotation, we sample from [-π, π] for yaw, pitch and row and then convert them to quaternions.

The Kuka LBR iiwa 14 R820 fixed-based manipulator has 7 joints that form a configuration. The primary difference of this robot from the 15 DOF planar manipulator is that the Kuka manipulator is 3D, which gives rise to much more complex swept volume geometries.

The Closed-loop robot has 10 joints connecting 8 linkages that are 0.5m long and 0.1m wide. The two base joints (marked as black cylinder in Figure 4) are fixed. The Y-shaped end-effector is rigid, and the tip of the effector (green) undergoes linear motion between the start (magenta) and end (blue). The start and end end-effector positions are sampled uniform-randomly in the yellow region. Invalid swept volume samples are rejected, this includes start and end positions with no valid inverse kinematic solution or solutions that result in self-collision. The 10 joint angles form a configuration. The $\mathcal{D}_{\text{we}}^{\text{sv}}$ weights listed in Table 1 correspond to joints in the clock-wise order starting from the upper base joint.

The Prismatic robot has 4 revolute DOFs and 4 prismatic DOFs. The black and yellow links are 0.5m long and 0.1m wide. The prismatic joints allow yellow links to retract inside black links up to 0.5 m. The revolute joint angle and prismatic joint length form a configuration.

The Youbot mobile manipulator is manufactured by the Kuka company. The mechanical wheels allow the robot to translate in any direction on the horizontal plane. The 5 joint angles of the manipulator along with the x-y position and yaw of the robot form a configuration. During training, the translation DOFs are limited to a 1.5m square.

The Parallel robot has 15 joints articulating three two-link legs connecting a fixed base (bottom hexagonal platform) to an end-effector (top hexagonal platform) with varying position and orientation. The legs each have three orthogonal joints on the fixed base and two orthogonal joints at the leg knee, a twist joint on the knee was not included because it is unaffected by inverse kinematics. The 15 joint angles form a configuration. The start and end end-effector position and orientation are sampled uniformly-randomly. Invalid swept volume samples are rejected, meaning that the start or end configuration has no valid inverse kinematic solution.

12.3 Implementation Details

In Sections 6 to 8, the DNNs used to learn $|\widetilde{\mathcal{SV}}_0|$ share the same hyper-parameters. These include: the number of neurons in the hidden layers = [1024, 512, 256], learning rate = 0.1, training batch size = 100 and the number of training epochs = 500 (the number of times the network utilizes the entire training dataset during training). A stochastic gradient descent-based optimizer is used by both the single layer networks and the DNNs. One hundred intermediate configurations are generated to compute $|\widetilde{\mathcal{SV}}_0|$. The octree used to compute swept volume measure ($|\widetilde{\mathcal{SV}}_0|$) has a resolution of Δ = 0.025m. The DNNs are trained with Tensorflow 1.6 on an Intel i7-6820HQ at 2.7GHz with 16GB of RAM. The training data generation is implemented within the open-source V-REP robot simulator platform. The performance of the network was evaluated by an evaluation dataset with ten thousand samples. This dataset was generated in the same fashion as the training data, but it was previously unseen by the network.

In Section 9, the DNN hyper-parameters include: the number of hidden neurons with ReLu activation = [500, 1000, 2000, 4000], learning rate = 0.0001, training batch size = 100 and the training terminates after 139 epochs. The output layer has 8000 neurons, representing the probability of occupation of each voxel. A stochastic gradient descend-based optimizer is used. The voxel grid used to approximate the swept volume geometry ($\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2)$) has a resolution Δ' = 0.1m and has 20x20x20 voxels in total. Ninety thousand training and ten thousand evaluation samples were generated. The computation time of the estimator is evaluated by averaging over 10,000 estimates with various batch sizes. The DNNs are also trained with Tensorflow 1.13 on the same computer. The training data was also generated using V-REP.

In Sections 6.2 and 7, motion planning and $|\widetilde{\mathcal{SV}}_0|$ computation time experiments were conducted using OMPL

in C++ on the same computer. Parameters of RRT other than ones mentioned in main paper are set to the default values in OMPL. This means an extend step size of 0.2 times the maximum Euclidean distance of any pair of points in C-space and a goal bias of 0.05. The V-REP platform is used to simulate the robot and collision detection. All planning was repeated 20 times.

References

Abrams S and Allen PK (2000) Computing swept volumes. *The Journal of Visualization and Computer Animation* 11(2): 69–82.

Amato NM, Bayazit OB, Dale LK, Jones C and Vallejo D (1998) Choosing good distance metrics and local planners for probabilistic roadmap methods. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. pp. 630–637.

Brin S (1995) Near neighbor search in large metric spaces. In: *Proc. of Int. Conf. on Very Large Data Bases*. pp. 574–584.

Campen M and Kobbelt L (2010) Polygonal boundary evaluation of Minkowski sums and swept volumes. In: *Computer Graphics Forum*, volume 29. pp. 1613–1622.

Chiang HTL, Faust A, Sugaya S and Tapia L (2018) Fast swept volume estimation with deep learning. In: Morales M, Tapia L, Sanchez-Ante G and Hutchinson S (eds.) *Algorithmic Foundations of Robotics XIII*. Springer, p. In print.

Ekenna C, Uwacu D, Thomas S and Amato NM (2015) Improved roadmap connection via local learning for sampling based planners. In: *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*. pp. 3227–3234.

Elbanhawi M and Simic M (2014) Sampling-based robot motion planning: A review. *IEEE Access* 2: 56–77.

Faust A, Oslund K, Ramirez O, Francis A, Tapia L, Fiser M and Davidson J (2018) PRM-RL: Long-range robotic navigation tasks by combining reinforcement learning and sampling-based planning. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. pp. 5113–5120.

Gaschler A, Petrick R, Kröger T, Khatib O and Knoll A (2013) Robot task and motion planning with sets of convex polyhedra. In: *Proc. Robotics: Sci. Sys. (RSS)*.

Goodfellow I, Bengio Y and Courville A (2016) *Deep learning*. MIT press.

Hahnloser RH, Sarpeshkar R, Mahowald MA, Douglas RJ and Seung HS (2000) Digital selection and analogue amplification coexist in a cortex-inspired silicon circuit. *Nature* 405(6789): 947–951.

Himmelstein JC, Ferre E and Laumond JP (2010) Swept volume approximation of polygon soups. *IEEE Trans. on Autom. Sci. and Eng.* 7(1): 177–183.

Hornik K (1991) Approximation capabilities of multilayer feedforward networks. *Neural Networks* 4(2): 251–257.

Ichter B, Harrison J and Pavone M (2018) Learning sampling distributions for robot motion planning. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. pp. 7087–7094.

Kaelbling LP and Lozano-Pérez T (2010) Hierarchical planning in the now. In: *Proc. Int. Conf. Artif. Intel.* pp. 33–42.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *Int. J. Robot. Res.* 30(7): 846–894.

Kavraki LE, Svestka P, Latombe JC and Overmars MH (1996) Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Trans. Robot.* 12(4): 566–580.

Kim YJ, Varadhan G, Lin MC and Manocha D (2004) Fast swept volume approximation of complex polyhedral models. *Computer-Aided Design* 36(11): 1013–1027.

Kingma D and Ba J (2014) Adam: A method for stochastic optimization. *International Conference on Learning Representations* .

Kuffner JJ (2004) Effective sampling and distance metrics for 3D rigid body path planning. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. pp. 3993–3998.

Kuffner JJ and LaValle SM (2000) RRT-Connect: An efficient approach to single-query path planning. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2. pp. 995–1001.

LaValle SM (2006) *Planning algorithms*. Cambridge university press.

LaValle SM and Kuffner JJ (2001) Randomized kinodynamic planning. *Int. J. Robot. Res.* 20(5): 378–400.

Lozano-Perez T (1990) Spatial planning: A configuration space approach. In: *Auto. Robot Vehicles*. pp. 259–271.

Mamou K and Ghorbel F (2009) A simple and efficient approach for 3D mesh approximate convex decomposition. In: *Int. Conf. on Image Processing (ICIP)*. IEEE, pp. 3501–3504.

McMahon T, Jacobs S, Boyd B, Tapia L and Amato NM (2012) Local randomization in neighbor selection improves PRM roadmap quality. In: *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*. pp. 4441–4448.

Palmieri L and Arras KO (2015) Distance metric learning for RRT-based motion planning with constant-time inference. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*. pp. 637–643.

Perrin N, Stasse O, Baudouin L, Lamiraux F and Yoshida E (2012) Fast humanoid robot collision-free footstep planning using swept volume approximations. *IEEE Trans. Robot.* 28(2): 427–439.

Schymura D (2014) An upper bound on the volume of the symmetric difference of a body and a congruent copy. *Advances in Geometry* 14(2): 287–298.

Şucan IA, Moll M and Kavraki LE (2012) The Open Motion Planning Library. *IEEE Robot. Automat. Mag.* 19(4): 72–82.

Völz A and Graichen K (2016) Distance metrics for path planning with dynamic roadmaps. In: *Proc. Int. Symp. on Robotics*. pp. 126–132.

Von Dziegielewski A, Hemmer M and Schömer E (2015) High precision conservative surface mesh generation for swept volumes. *IEEE Trans. on Autom. Sci. and Eng.* 12(1): 183–191.

Wampler CW (1986) Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *Trans. on Sys., Man, and Cybern.* 16(1): 93–101.

Wolfslag WJ, Bharatheesha M, Moerland TM and Wisse M (2018) RRT-CoLearn: towards kinodynamic planning without numerical trajectory optimization. *Robot. and Automat. Lett.* 3(3): 1655–1662.

Xavier PG (1997) Fast swept-volume distance for robust collision detection. In: *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, volume 2. IEEE, pp. 1162–1169.