# Multitask and Transfer Learning of Geometric Robot Motion

Satomi Sugaya[1], Mohammad R. Yousefi[1], Andrew R. Ferdinand[3,4], Marco Morales[2], and Lydia Tapia[1]

***Abstract*— When a learning solution is needed for different robots, a model is often trained for each robot geometry, even if the robotic task is the same and the robots are structurally similar. In this paper, we address the problem of transfer learning of swept volume predictors for the motion of articulated robots with similar geometric structure. The swept volume is a scalar value corresponding to the space occupied by an entire motion of the robot. Swept volume has many applications, including being an ideal distance measure for sampling based motion planners, but it is expensive to compute. We address this learning problem through a multitask network where a common input is used to learn multiple related tasks. In this work a single network learns the kinematic-geometric information common among robots. In order to identify the properties of our multitask network favorable for transfer, we evaluate transfer properties of several shared layers, number of robots in multitask training, and feature layers. We demonstrate positive transfer results with a training set that is a fraction of the data size used in the multitask and baseline training. All the robots considered are 7-DOF manipulators with links with a variety of lengths and shapes. We also present a study of the weights and activations of the trained networks that show high correlation with the transferability patterns we observed.**

## I. INTRODUCTION

Robotic geometry is critical for successful task completion, as even the simplest essential tasks include determining the space that a robot occupies or whether it hits any obstacles at specific configurations [1]. Similar end-use robotic systems can have comparable geometries. For example, different robot manipulators have similar geometric designs, with differences in the types of joints or the size of links. Existing research has focused on dynamic and kinematic similarities [2]–[4] but the geometric similarities are not often considered when robot tasks are learned despite the conservation of form. Rather, a learned model is often developed for each robot geometry, at great computational cost for learning or the collection of training data.

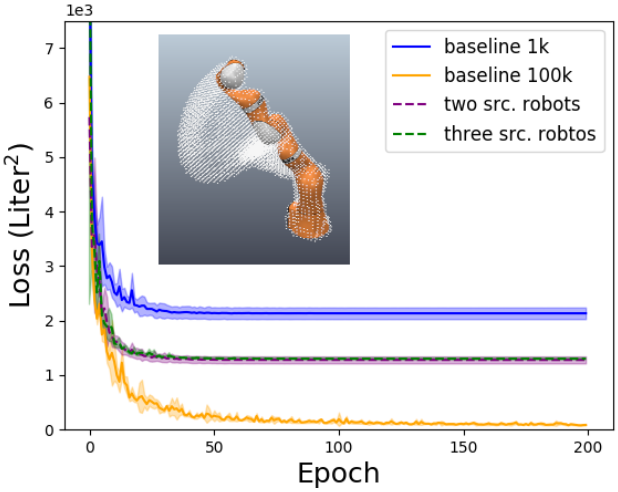


Fig. 1. Learning curves of $\widetilde{\mathcal{SV}}$ training for Kuka. Evaluation loss (in liter$^2$) vs Epoch. Baselines with 1000 (blue) and 100k (orange) data samples, and transfer training with 1000 samples using source MT-Net trained with two source robots (dashed purple) and three source robots (dashed green). The inset shows Kuka and its swept volume as it moves through space.

Learning how to train robots to perform new tasks and applying learned models to new problems has been explored in transfer learning [5], [6]. It has been used, for example, where training data is scarce [3], [7], or to mitigate risks to robots [8]. Transfer learning can improve learning quality of a task (target task) by exploiting relevant knowledge acquired previously from learning another related task (source task). For example, [4] shows that kinematic and dynamics models can be transfer-learned between different robots via principle component analysis and manifold alignment.

Our prior work has shown that trained deep neural networks (DNNs) can approximate the geometric measure of Swept Volume (SV) [9]–[11], *i.e.*, computing the volume in space occupied during *an entire motion* of an object (Figure 1, inset). SV has many practical uses in machining verification (validation of a machining process), geometric modeling, maintainability study, virtual assembly, and task and motion planning [12]–[16]. It is also considered to be an ideal distance measure for Sampling-Based Robotic Motion Planning [17]. The SV scalar function, $\mathcal{SV}$, that maps robot configurations to the corresponding SV is a highly non-linear function with many numerical approximations [12]–

* The authors thank John Baxter for his help generating data. This work was conducted as part of the University of New Mexico Women in STEM Faculty Development Fund. Sugaya and Tapia are partially supported by the National Science Foundation under Grant Number 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. Sugaya, Yousefi, Ferdinand, and Tapia 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. Morales is partially supported by Academia Mexicana de Cultura A.C.

[1]Department of Computer Science, University of New Mexico, MSC01 11301 University of New Mexico, Albuquerque, NM 87131, USA. satomi@unm.edu, myousefi@unm.edu, and (corresponding) tapia@cs.unm.edu

[2] Department of Computer Science, Instituto Tecnológico Autónomo de México, Mexico City, 01080, México marco.morales@itam.mx, and University of Illinois at Urbana-Champaign, Urbana, IL, 61801, USA moralesa@illinois.edu

[3]Department of Electrical and Computer Engineering, University of New Mexico, MSC01 11301 University of New Mexico, Albuquerque, NM 87131, USA.

[4]Present address: Department of Physics, University of Colorado Boulder, Boulder, CO 80309, USA. andrew.ferdinand@colorado.edu

[14], [16], [18] which are computationally costly. While our prior work found that a fast and accurate approximation of SV is possible, [9]–[11], the training data generation was time-costly. For example, it took about one week to generate 100k samples of SV data per robot, an amount required for high accuracy training [10]!

In this paper we apply transfer learning to the learning of a SV function approximator, $\widetilde{\mathcal{SV}}$, using deep learning. This study aims to gain insights on how the network is learning and whether the network captures the geometrical similarities among robots with similar structures (four manipulators with 7 degrees of freedom). Our approach is posed within the context of multitask learning [19], [20] where multiple, distinct but related, tasks are learned from a common input. In this work, multitask learning extracts parts of DNN structures, *i.e.* feature mappings, that are shared among distinct robotic manipulator arms. The feature mappings are then adopted as parts of the network initialization of the DNN for a new robot trained with a reduced portion of the data set. Accuracy improvements of the transfer-learned SV function approximators are shown.

Through the exploration of transfer learning for the robot set, Kuka LBR iiwa 14 R820 fixed-base manipulator, Rethink Robotics' Baxter and Sawyer robot arms, and Barrett WAM, we present some insightful results that better elucidate transferred outcomes. Specifically, this work contributes:

- A positive transfer learning result for SV function approximators between different robots using a fraction of the training data set used in the multitask and baseline training.
- An evaluation of multitask-network shared layers and their feature transferability.
- A study of the DNNs' layer outputs and activations that provide insight into the transferability patterns observed in the experiments.

## II. Related Work

### A. Robot Swept Volume Computation: Numerical and Learning Approaches

There are largely three classes of modern algorithms that compute approximations of swept volumes, SVs, of geometrically complex objects. One class is the boundary-based methods that compute an outer boundary surface of the SV. The surface is computed as the union of the surfaces resulting from sweeping and rotational motions of the object [12], [13], [18]. These methods comput the boundaries of moving solids are especially slow for articulated bodies. Another class of algorithm is the convex polyhedra-based method. In these methods, convex polyherdons are used to decompose robot models at each motion step. The SV is computed as the union of convex hulls [14], [16] approximating the robot body motion. The third class is the occupancy grid-based methods in which the work space is voxelized. The voxels touched by a robot motion are recorded as the SV [21], [22].

In our previous learning approach [9], [10], the SV scalar function, $\mathcal{SV}(\boldsymbol{c}_1, \boldsymbol{c}_2)$, that maps the initial ($\boldsymbol{c}_1$) and final ($\boldsymbol{c}_2$) robot configurations to the corresponding SV has been shown to be Lipschitz continuous. This continuity property enables fully-connected feed-forward DNNs to approximate $\mathcal{SV}$ [23]. The DNNs, trained via supervised learning, have been shown to predict SVs fast and accurately. In addition to the scalar value, SV *geometry* has also been learned [11]. When considering learning the geometry of SV given by a voxelized representation, the learning problem is no longer of continuous function approximation but, rather, of classification. The learning problem has been formulated to predict SV geometry as a mapping from $(\boldsymbol{c}_1, \boldsymbol{c}_2)$ to on/off voxels in the voxelized space. This mapping is learned with high accuracy for robots with a variety of joint properties and degrees-of-freedom. The scalar and geometry predictors are shown to be faster than the occupancy grid-based methods [21], [22]. Our work builds on [9], [10] to investigate transfer learning of $\widetilde{\mathcal{SV}}$s.

### B. Multitask Learning and Transfer Learning Geometric Properties of Robotic Manipulator Arms

Large bodies of literature exist in robotics regarding transfer learning and multitask learning. For example, they have been used in reinforcement learning [24], [25], simulation-to-real transfer [7], [26], sensor information processing [7], [27]–[30], learning multiple tasks and transfer of robot models, kinematics, and dynamics [2], [3], [31]–[33].

The deep multitask learning method employed in this work is found ubiquitously in robotics and beyond [34], [35]. Specifically, our network structure falls under the shared trunk category [35] where information common to different tasks are learned by shared lower network layers (hard parameter sharing [34]), and the specificity of each task by task-specific top layers (see Figure 2). A convolutional network with such structure is used in [36] to learn grasping, pushing, and poking robotic tasks from image inputs. Similar use of convolutional networks are seen widely in visual tasks from semantic localization and odometry [37], segmentation and attention [38], to pose estimation and action detection [39]. One of the earliest works using the shared trunk network structure learns multiple steering tasks for autonomous driving from a visual input using a single layer network [40]. In contrast, the task we consider does not require visual or other sensory information, but a pair of robot configurations to map robot kinematic and geometric properties to a scalar that quantifies the amount of motion. This much simpler input structure enables us to use a much simpler network model.

The transfer learning performed in this paper exploits the similarities in different manipulator arms of joint-kinematics and link-geometry relationships, *i.e.*, kinematic models. In developmental robotics, a kinematic model learned by an experienced robot using its sensory information is transferred to a novice in the form of data generated by the experienced robot [2]. Similarly, transfer learning is used to accelerate the learning of inverse dynamics models of manipulator arms in [3]. A transfer learning framework for forward and inverse kinematics, inverse dynamics, and operational space

control between different robot models [4] enables transfer by finding common representations in the data space, followed by transformation between them. In humanoid robots and animation, the self-collision free pose transfer (motion re-targeting) problem is addressed by learning a common latent space during separate learning of safe and unsafe poses among a variety of subjects [41]. Our aim is not to learn manipulation tasks [2]–[4], but to learn how the kinematic-geometric relationships affects the amount of robot motion. Additionally, our simple method of hard-parameter sharing [34] and fine-tuning bypasses the difficulties of learning relational mappings between data or models [4], [41].

## III. Preliminary: Learning $\mathcal{SV}$

We use the same learning problem formulation of $\mathcal{SV}$ as in [9] to conduct our multitask and transfer learning experiments. Given a pair of robot configurations $\boldsymbol{c}_1, \boldsymbol{c}_2 \in \mathbb{R}^d$, where $d$ is the Degrees of Freedom (DOF) of a robot, the SV scalar function corresponding to a straight-line trajectory in the configuration space is given by

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

where $t \in [0,1]$ is a time parameter that maps the points along the trajectory so that it starts at $\boldsymbol{c}_1$ at $t=0$ and finishes at $\boldsymbol{c}_2$ at $t=1$. $\mathcal{V}$ maps the intermediate configurations to their corresponding workspace volume.

The continuous function approximator, represented by $\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2|\boldsymbol{\theta})$, is a DNN trained using supervised learning. The training objective is to find the parameters $\boldsymbol{\theta}$ that minimize the loss $\mathcal{L}$:

$$\tilde{\boldsymbol{\theta}} = \arg\min_{\boldsymbol{\theta}} \mathcal{L}\left(\widetilde{\mathcal{SV}}(\boldsymbol{c}_1, \boldsymbol{c}_2|\boldsymbol{\theta}), \mathcal{SV}\left(\boldsymbol{c}_1, \boldsymbol{c}_2\right)\right). \qquad (2)$$

We consider a bottom-heavy, feed-forward fully connected DNN similar to [9] with three hidden layers that have $[1024, 512, 256]$ neurons from bottom to top. Neurons use ReLU activation except for the output neurons which use no activation (identity) function. The input features are a one-dimensional vector holding the values of $\boldsymbol{c}_1$ and $\boldsymbol{c}_2$ and the output is a single scalar SV prediction. For ease of presentation, we let $\boldsymbol{W}^{i,i+1}$ be the weight matrix, $\boldsymbol{b}^{i,i+1}$ be the bias vector, and $\Phi_{i+1}(\cdot)$ be the activation function that map features at layer $i$ to layer $i+1$. For our network, $i = 0, 1, 2, 3$ with $i=0$ being the input layer and $i=4$ the output layer. As such, the output at the $(i+1)$-th layer expressed in terms of the output from the previous layer, $\boldsymbol{x}_i$ is

$$\Phi_{i+1}\left(\boldsymbol{W}^{i,i+1}\boldsymbol{x}_i + \boldsymbol{b}^{i,i+1}\right), \quad \boldsymbol{x}_0 = (\boldsymbol{c}_1, \boldsymbol{c}_2). \qquad (3)$$

Previous studies show that the learning of $\widetilde{\mathcal{SV}}$ is stable and the learned predictors are highly accurate with mid to high 90% accuracy when trained with 100k data samples [10]. Using transfer learning in this work, we leverage these properties of $\widetilde{\mathcal{SV}}$ to improve upon the learning of $\widetilde{\mathcal{SV}}$ with *reduced*, *e.g.*, a factor of 1000 fewer, training data for *new* similar robots.

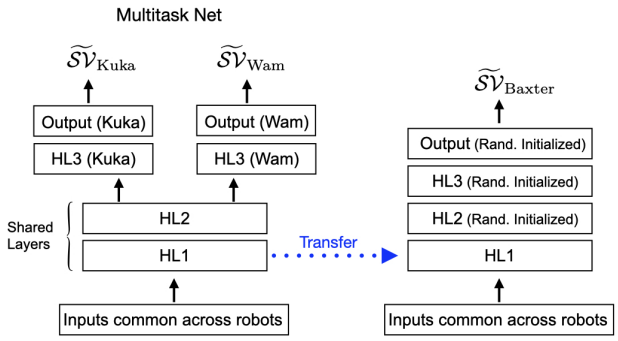


Fig. 2. An MT-Net (left) trained to learn $\widetilde{\mathcal{SV}}$ with Kuka and WAM simultaneously. The first (HL1) and second (HL2) hidden layers are shared between the robots. The transfer net (right) learning $\widetilde{\mathcal{SV}}$ for Baxter. The HL1 features of the MT-Net is shown to be transferred.

## IV. Transfer Learning Setup

We design experiments to compute $\widetilde{\mathcal{SV}}$ for a new robot geometry based on a previously learned $\widetilde{\mathcal{SV}}$ from robots with different geometries. We evaluate transferability through accuracy, computation time, and the dataset size required for training.

### A. Simultaneous Learning of Multiple $\widetilde{\mathcal{SV}}$s

Our multitask-$\widetilde{\mathcal{SV}}$ network (MT-Net) is a single multitask SV function approximator shared across multiple *source robots* with the aim of learning common features for these robots. The multiple tasks correspond to the learning of distinct swept volume functions of the robots trained together. Figure 2 shows a MT-Net trained on two robots, Kuka and WAM. It has a shared trunk structure [35] where the first and second hidden layers, HL1 and HL2, are shared between Kuka and WAM. The output features of HL2 are connected to a separate task-specific hidden layer (HL3) and an output layer for each robot. This structure leads the network to learn the common features in the shared layers and the robot-specific features in the task-specific layers. We use the GradNorm Algorithm [42] to determine the learning parameters and balance the gradients and learning rates across the different tasks.

### B. Transferring Shared Layer Features

All the robots we consider have 7 DOFs and similar kinematic-chain link structure. However, differences in joint types, link shapes, and masses produce different reachable workspace volumes and kinematic properties.

We transfer information in the shared layers of the MT-Net to boost the learning of $\widetilde{\mathcal{SV}}$ for new robots. The MT-Net, the entity from which information is transferred to another, is commonly referred to as the *source*. The receiving entity, the network for a new robot, is termed the *target*. An example of this is depicted in Figure 2 where the target network learning $\widetilde{\mathcal{SV}}$ for the Baxter robot is shown on the right. Rather than starting with fully randomized weights, HL1 weights of the MT-Net trained with Kuka and WAM are adopted. Networks, $\widetilde{\mathcal{SV}}$s, trained *without* this transfer of weights will be referred

| | $J_0$ | $J_1$ | $J_2$ | $J_3$ | $J_4$ | $J_5$ | $J_6$ |
|---|---|---|---|---|---|---|---|
| Minimum | $-97.5°$ | $-114.6°$ | $-160.5°$ | $-2.865°$ | $-170°$ | $-90°$ | $-171.9°$ |
| Maximum | $97.5°$ | $60°$ | $160.5°$ | $120°$ | $74.5°$ | $91.63°$ | $171.9°$ |

TABLE I
COMMON JOINT LIMITS FOR KUKA, WAM, BAXTER AND SAWYER ROBOTS.

to as the baselines. Additionally, we evaluate the effect of transferring different hidden layers on the learning outcome compared to the baseline (Sec. V-B).

### C. Training Data Generation and Network Training

The swept volume data, SV, used for learning is generated using an occupancy grid-based method [22]. Two constraints are applied. First, robot joint limits are constrained to a common range, $J_i$, which is defined as the intersection of the joint limits for Kuka, WAM, Baxter and Sawyer (Table I). This limitation ensures that generated configurations are valid for all four robots. Second, we ensure that the SV values for each robot are computed from the same set of configuration pairs. The configuration pairs are drawn from the ranges in Table I using a seeded uniform pseudo-random number generator. The set includes 100k samples for training and 10k samples for evaluation for each robot. As such, by assuring that all the robots have the same input data, we constrain the otherwise multi-modal multitask learning (when *both* the inputs and the labels are distinct for the robots) to a multitask framework.

The baseline, multitask, and transfer networks are trained in a supervised manner with the objective given in Eq. (2). Mean Square Error loss and the Adagrad optimizer are used with an initial learning rate of 0.1. The baseline and transfer network structures are as described in Sec. III. This same structure is used in the multitask network for each source robot. The $\alpha$ parameter of the GradNorm algorithm [42] is set to unity. The mini-batch sizes of 1, 10, and 1000 are used for training data sizes of 100, 1000, and 100k, respectively, to keep the number of stochastic gradients runs for a single epoch constant. All the labeled data is (arbitrarily) normalized to Kuka's maximum volume of 311.17 liters.

Data generation and training has been done using Intel(R) Xeon(R) E-2146G CPU @ 3.50GHz with 32GB of memory and Nvidia Quadro P1000 GPU.

## V. ANALYSIS AND RESULTS

### A. Transfer Gain

We quantify the transfer gain by measuring the learning speedup and accuracy improvement of $\widetilde{\mathcal{SV}}$. This is done by analyzing the accuracy curve. We characterize the accuracy by the fitted model accuracy $= a\exp^{-b\,\text{epoch}} + c$ parameterized by constants $a, b$ and $c$. To quantify speedup, we query the accuracy models to find the epoch at which the fitted model reaches a specified accuracy threshold. The accuracy gain is quantified using the asymptotic values of the fitted accuracy curves. Figure 3 illustrates the speedup

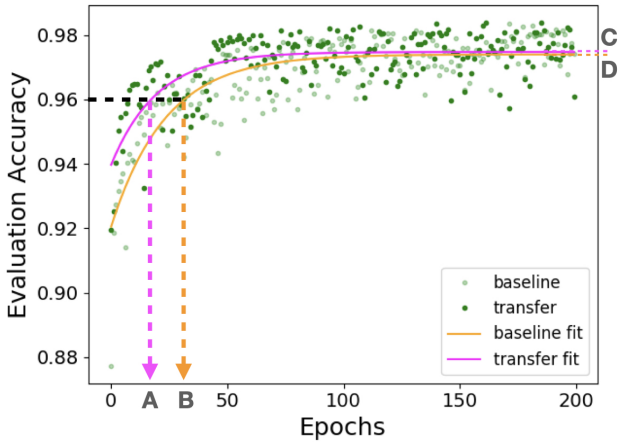


Fig. 3. Evaluation accuracy for WAM during baseline (transparent green) and transfer (green) training along with corresponding best-fit curves in magenta and orange.

and accuracy gain for a transfer experiment. The speedup for a 96% accuracy threshold is computed by the difference in corresponding epochs for transfer (A) and baseline (B). The quantity $A-B$ is the speedup due to transfer indicating faster $(+)$ or slower $(-)$. This measure is obtained for accuracy thresholds between 80% and 100%, in 1% increments. The asymptotic accuracy values for transfer (C) and baseline (D) are used to compute the accuracy gain due to transfer.

Two types of transfer learning experiments are performed. One assesses transferability properties of MT-Net and its features. Through this experiment, we select a source MT-Net and features best suited to transfer. The other experiment uses the selected source to perform transfer learning with reduced training data sizes.

### B. Multitask Net and Feature Transferability Assessment

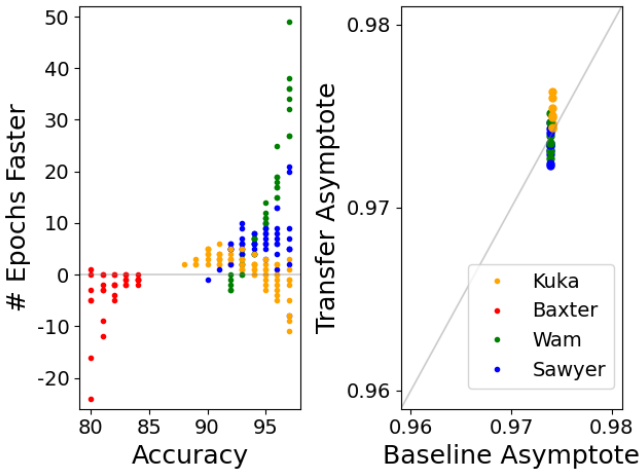


Fig. 4. Speedup (left) and accuracy gain (right) results for transferring the first HL of the MT-Net trained using two robots with one SL. This data is shown for *unseen* target robots: Kuka (orange), Baxter (red), WAM (green), and Sawyer (blue).

A transfer experiment is defined by four parameters: (1) number of source robots, (2) number of layers in the shared

| Trf. Feature | | 1st HL | | 2nd HL | | 3rd HL | |
|---|---|---|---|---|---|---|---|
| # Src. Robots | | 2 | 3 | 2 | 3 | 2 | 3 |
| Speedup (%) | 1 SL | 74 | 87 | – | – | – | – |
| | 2 SL | 63 | 66 | 23 | 38 | – | – |
| | 3 SL | 71 | 75 | 17 | 33 | 47 | 33 |
| Accuracy Gain (%) | 1 SL | 83 | 75 | – | – | – | – |
| | 2 SL | 58 | 23 | 23 | 23 | – | – |
| | 3 SL | 60 | 23 | 20 | 23 | 75 | 50 |

TABLE II

TRANSFERABILITY MEASURE, PERCENTAGE OF POSITIVE-TRANSFER OCCURRENCES, FOR EACH TRIPLET. THE HL PARAMETER VALUES ARE SHOWN IN THE COLUMNS, EACH FURTHER DIVIDED INTO THE SOURCE ROBOT PARAMETER VALUES. THE SL PARAMETER VALUES ARE GIVEN IN THE ROWS, FOR SPEEDUP AND ACCURACY GAIN.

| Data Size | | 100 | | 1000 | |
|---|---|---|---|---|---|
| # Source Robots | | 2 | 3 | 2 | 3 |
| Transferability (%) | Speedup | 100 | 100 | 94 | 100 |
| | Accuracy | 83 | 100 | 92 | 100 |
| Average Gain | Speedup | 2.2 | 3.5 | 1 | 1.2 |
| | Accuracy | 4.8 | 5.9 | 1.5 | 1.5 |

TABLE III

TRANSFERABILITY MEASURE (% IMPROVEMENT, FIRST TWO ROWS), AVERAGE SPEEDUP (EPOCHS, THIRD ROW), AND ACCURACY GAIN (% IMPROVEMENT, FOURTH ROW) OF TRANSFER LEARNING $\widetilde{\mathcal{SV}}$ WITH REDUCED TRAINING SAMPLES. THE RESULTS FOR 100 AND 1000 SAMPLES ARE GIVEN IN THE TWO COLUMNS WITH EACH COLUMN FURTHER DIVIDED INTO TWO COLUMNS FOR TWO AND THREE SOURCE ROBOTS. THE SOURCE MT-NETS WERE TRAINED WITH ONE SL.

trunk (SL), (3) the transferred hidden layer weights (HL) and (4) a target robot. Parameters 1, 2 and 3 are collectively referred to as a triplet. We systematically perform a transfer experiment with full data (100k) for each unique combination of triplet and an unseen target to assess the triplet transferability. Triplets with the best transferability are selected for the use in the transfer experiments with reduced data sizes. We quantify the tranferability as the fraction of experiments that result in positive values for speedup and accuracy gain (as defined in Sec. V-A). The summary of the triplet transferability averaged over target robots is reported in Table II. The parameters defining the source triplet are given in the columns and rows of the table. The three columns in the table correspond to the first, second, and third HL transferred. These columns are further divided into two columns each, corresponding to the MT-Net trained with two and three source robots. The number of SL trained together are given in three rows each for the speedup and accuracy gain measures.

The triplets [two source robots, one SL, first HL] and [three source robots, one SL, first HL] have the highest transferability at 74% and 87% for speedup and 83% and 75% accuracy gain, respectively. Figure 4 shows the speedup (left) and accuracy gain (right) for the [two source robots, one SL, first HL] triplet. Orange, red, green, and blue data points correspond respectively to unseen targets, Kuka, Baxter, WAM, and Sawyer. The positive transfer occurrences are instances above the zero line on the left and the diagonal line on the right figures. This experiment demonstrates that multitask learning is beneficial for extracting relevant kinematic and geometric information common to the robots in the form of feature mappings.

### *C. Transfer Learning for Learning $\widetilde{\mathcal{SV}}$ with Reduced Samples*

We perform the transfer experiment with reduced data sizes using the MT-Net characterized by the best performing triplets selected in V-B. The source MT-Nets are transferred to learn the deep swept volume function approximator, $\widetilde{\mathcal{SV}}$, of unseen target robots using one thousandth (100) and one hundredth (1000) of the original (100k) samples.

| Average Prediction Accuracy (%) | | | | | | |
|---|---|---|---|---|---|---|
| Data Size | 100 | | 1000 | | 100k | |
| Baseline | 73.8 | | 88.1 | | 92.7 | |
| # Src. Robots | 2 | 3 | 2 | 3 | 2 | 3 |
| Transfer | 77.6 | 78.8 | 89.7 | 89.7 | 97.8 | 97.8 |

TABLE IV

AVERAGE PREDICTION ACCURACY OF $\widetilde{\mathcal{SV}}$ TRAINED WITH 100, 1000, AND 100K DATA SAMPLES. FOR THE BASELINE, THE AVERAGE IS TAKEN ACROSS ALL ROBOTS. FOR TRANSFER, THE AVERAGE IS TAKEN OVER UNSEEN TARGETS. SOURCE MT-NETS WERE TRAINED USING TWO AND THREE SOURCE ROBOTS WITH ONE SL.

Figure 1 shows loss curves for Kuka baseline training with 1000 (blue) and 100k (orange) data samples. Also shown are transfer training loss curves with 1000 samples using source MT-Net trained with two (dashed purple) and three (dashed green) robots. As expected, baseline training with reduced data samples result in a much higher loss (blue), $\sim 2\times10^3$ liter$^2$, compared to training with full data (orange), $< 1\times10^3$ liter$^2$. The benefit of transfer learning is seen in two ways. First, convergence is seen at earlier epochs in the transfer curves (dash purple and green) compared to the baseline with 1000 samples (blue). Second, the transfer curves converge to a lower loss value compared to the same baseline.

Next, we assess the transferability measure and average transfer gains for each of the transfer experiments and find that the transfer learning conducted was advantageous for learning more accurate $\widetilde{\mathcal{SV}}$ when training with reduced data sizes. Table III summarizes the transferability measure (%, first two rows), average speedup (epochs, third row), and accuracy gain (%, fourth row) of this transfer learning experiment. The results for 100 and 1000 samples are given in the two columns with each column further divided into two columns for two and three source robots. For 100 data samples, the transferability is 83% for accuracy of two source robot MT-Nets and 100% in all other cases. Speedup and accuracy transferability are 94% and 92% for two source robot and 100% for three source robot MT-Nets with 1000 samples.

While on average there is a speedup gain of 1 to 3.5 epochs (Table III, third row), the amount of data required to achieve a high accuracy is reduced by up to *a thousand-fold* using the transfer network. Specifically, as shown in Table IV, an accuracy of 89.7% can be achieved with 1000 samples (third and fourth columns). This is 97% of the accuracy found by presenting a hundred-fold larger dataset (100k, last two columns). These results correspond to and are reflected in the loss curve trends in Figure 1. Even more surprisingly, the transfer network achieves an accuracy of 78.8%, *i.e.*, 85% of the accuracy of the 100k dataset, with a thousand-fold fewer data samples. Comparing the transfer gains for 100 and 1000 data samples (Tables III and IV), our transfer learning paradigm benefits learning of $\widetilde{\mathcal{SV}}$ with reduced data size. The average accuracy gains yielded are higher for 100 training samples at 4.8% and 5.9% compared to 1.5% for 1000 samples (Table III, fourth row). The gains results in the improvement of predictor accuracy from 73.8% to 78.8% and from 88.1% to 89.7% on average when training with 100 and 1000 samples respectively (Table IV).

Additionally, this transfer learning paradigm can yield a better predictor accuracy than the baseline trained with 100k samples when the same amount of data is used in the transfer training. The last two columns of Table IV shows the accuracy improvement from 92.7% to 97.8%. While this result requires a full data set, this is an important result for applications that require a more accurate predictor.

Comparing the average accuracy gains for 100 training samples between two and three source robots (Table III, columns two and three, fourth row) suggests that it is more effective to use source MT-Net trained with three robots in the transfer training. For example, using three source robots there's on average 5.9% accuracy over the baseline while there's only a gain of 4.8% using two source robots. Intuitively, a network trained with more related source robots learns the common properties of the tasks more effectively.

### *D. Feature Transferability and Insights from Network Analysis*

During the investigation of source MT-Net properties favorable for the transfer experiments described in Sec. V-B, we notice an intriguing pattern in the transferability of different feature layers. As noted in Table II, the first HL (columns one and two) and third HL (columns five and six) are more transferable than the second HL (columns three and four). Motivated by this, we analyze the baseline $\widetilde{\mathcal{SV}}$ networks for all robots. We conclude this paper with some preliminary results that may shed some light on the transferability patterns. We examine two network quantities: weights, $\boldsymbol{W}^{i,i+1}$, and feature-layer outputs, $\Phi_i$ (Eq. 3).

We analyze the weights on the assumption that $\boldsymbol{W}^{i,i+1}$'s can be treated as collections of random variables. This assumption is reasonable considering that elements of $\boldsymbol{W}^{i,i+1}$'s are are initialized randomly, and given the mechanism of stochastic gradient descent [43]. Figure 5 shows the probability density of weight values, *i.e.*, the elements of $\boldsymbol{W}^{i,i+1}$'s, for each robot. The density of weights for the first and output layers, $\boldsymbol{W}^{0,1}$ and $\boldsymbol{W}^{3,\text{out}}$, look almost Lorentzian with the weight values highly concentrated around the center. In contrast, weights between the first and second layers and between the second and third layers, $\boldsymbol{W}^{1,2}$ and $\boldsymbol{W}^{2,3}$, are similar and seem to approximate bounded uniform densities.

The striking similarity of distribution of weight densities among robots within each layer may be one of the underlying reasons why our transfer learning paradigm works. Furthermore, Baxter's densities show visible deviations from others for $\boldsymbol{W}^{0,1}$, $\boldsymbol{W}^{1,2}$, and $\boldsymbol{W}^{2,3}$. This could explain the trend of Baxter not benefiting from transfer learning as much as other target robots. An instance of this can be observed on the left panel of Figure 4 where Baxter's speedup is mostly negative (red data points).

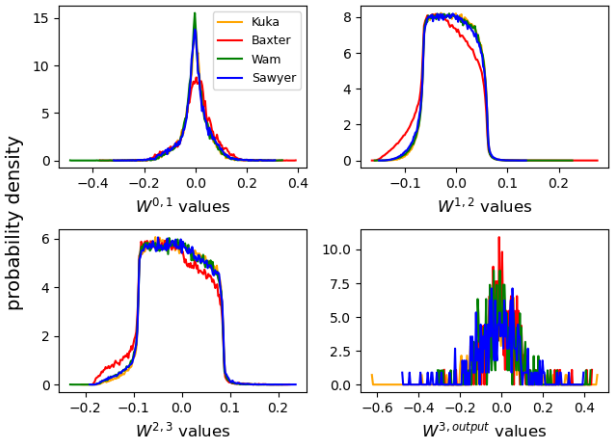


Fig. 5. Probability density of weight values at each layer for each robot. The densities for $\boldsymbol{W}^{0,1}$ are shown on upper left, $\boldsymbol{W}^{1,2}$ on upper right, $\boldsymbol{W}^{2,3}$ on lower left, and $\boldsymbol{W}^{3,\text{output}}$ on lower right.

| KS Test $p-value$ | | | | |
|---|---|---|---|---|
| $\boldsymbol{W}^{0,1}$: First hidden layer | | | | |
| | K | B | W | S |
| K | 1 | $7\times10^{-22}$ | $3\times10^{-1}$ | $1\times10^{-14}$ |
| B | $7\times10^{-22}$ | 1 | $1\times10^{-18}$ | $6\times10^{-9}$ |
| W | $3\times10^{-1}$ | $1\times10^{-18}$ | 1 | $2\times10^{-13}$ |
| S | $1\times10^{-14}$ | $6\times10^{-9}$ | $2\times10^{-13}$ | 1 |
| $\boldsymbol{W}^{1,2}$: Second hidden layer | | | | |
| K | 1 | 0 | $3\times10^{-13}$ | $5\times10^{-64}$ |
| B | 0 | 1 | 0 | $1\times10^{-107}$ |
| W | $3\times10^{-13}$ | 0 | 1 | $6\times10^{-125}$ |
| S | $5\times10^{-64}$ | $1\times10^{-107}$ | $6\times10^{-125}$ | 1 |
| $\boldsymbol{W}^{2,3}$: Third hidden Layer | | | | |
| K | 1 | $4\times10^{-27}$ | $3\times10^{-4}$ | $6\times10^{-6}$ |
| B | $4\times10^{-27}$ | 1 | $9\times10^{-15}$ | $1\times10^{-11}$ |
| W | $3\times10^{-4}$ | $9\times10^{-15}$ | 1 | $9\times10^{-2}$ |
| S | $6\times10^{-6}$ | $1\times10^{-11}$ | $9\times10^{-2}$ | 1 |

TABLE V

TWO-SAMPLE KS TEST P-VALUES OF THE HIDDEN LAYER WEIGHT DISTRIBUTIONS BETWEEN EACH ROBOT THE FIRST ($\boldsymbol{W}^{0,1}$), SECOND ($\boldsymbol{W}^{1,2}$), AND THIRD ($\boldsymbol{W}^{2,3}$) HIDDEN LAYERS.

We examine whether the similarities in weight distributions are correlated with the transferability trends observed

(first HL and third HL seem to be more transferable than second HL) using a two-sample Kolmogorov-Smirnov (KS) test [44]. We compare the weight distributions, *i.e.*, CDF of the densities, between robots for each layer through a two-sample KS test as shown in Table V. A two-sample KS test p-value is the probability that the two tested datasets were drawn from the same parent distribution. The p-values range from 0 to 1, where in our context p-value $= 0$ corresponds to the weight distributions being statistically distinct and p-value $= 1$ corresponds to the weight distributions being statistically identical.

These tests suggest a correlation between a layer's weight distribution and the observed transferability trends. First, the relatively low p-values for the second hidden layer weights ($\boldsymbol{W}^{1,2}$) suggest that the weight distributions in this layer are more tailored to each robot rather than the first ($\boldsymbol{W}^{0,1}$) and third hidden layers ($\boldsymbol{W}^{2,3}$). This may be why the second hidden layer performed poorly when transferred. A second observation is that the KS tests including Baxter result in consistently lower p-values. This indicate that Baxter's network weights deviate more significantly from the other robots' network's weight distributions. The third observation is the high p-values of $\mathcal{O}(0.1)$ in the Kuka-WAM $\boldsymbol{W}^{0,1}$ in the WAM-Sawyer $\boldsymbol{W}^{2,3}$ KS tests. These p-values indicate a higher degree of similarities in these distributions. Even with over $14,000$ and $130,000$ weight values in their respective distributions, statistically there remains an $\mathcal{O}(10\%)$ probability that the weights came from the same parent distributions. This degree of similarity may have contributed to the trends observed in our multi-task and transfer learning studies.

We additionally analyze feature layer outputs, $\Phi_i$s, at each layer. Our comparison of $\Phi_i$s is inspired by two questions. First, how different are the swept volume functions between robots to begin with? Second, will this difference correlate with transferability between source and target robots? Consider the swept volume functions for the Kuka and Baxter robots, $\mathcal{SV}_{\mathrm{K}}$ and $\mathcal{SV}_{\mathrm{B}}$. When analytic forms are available for the functions, the 2-norm [45] can be used as a proxy (technically, swept volume is not a metric [10]) for the difference between the $\mathcal{SV}$s:

$$\left( \int d\boldsymbol{c} |\mathcal{SV}_{\mathrm{K}} - \mathcal{SV}_{\mathrm{B}}|^2 \right)^{1/2}, \tag{4}$$

where $d\boldsymbol{c}$ is the differential elements in the configuration space. While we do not have the analytic forms, we *can* still compute relevant outputs, *i.e.*, successive values of $\mathcal{SV}$ for a given interval in order to approximate the integral. Therefore, a proxy to the 2-norm may be computed by

$$\left( \sum_n |\mathcal{SV}_{\mathrm{K},n} - \mathcal{SV}_{\mathrm{B},n}|^2 \right)^{1/2} \quad n \in [a,b]_j,\ j = 1,\dots,7, \tag{5}$$

where $[a,b]$ are the minimum and maximum joint-angles for j-th joint common to all robots. The index $n$ enumerates the monotonically increasing values of the joint angles with an arbitrary increment. We generate such a SV data set with 200 increments for each robot. This distance proxy is computed between Kuka and each robot: 2.3 L for WAM, 8.0 L for Sawyer, and 25.2 L for Baxter. Again, Baxter shows the largest difference.

We, *exploratorily*, apply an analogous distance proxy to compare each feature layer between Kuka and other robots. Although this is an arbitrary measure to be applied to the feature outputs, the distance proxy quantifies the neuron-wise difference of the cumulative activation over the successive robot motion for $[a,b]_j, j \in (1,\dots,7)$ between a given pair of robots. In other words, this proxy computes the cumulative difference in the amount of neuron-wise activation between two robots over the 200 motion steps from the initial to the final configurations. The question then becomes which neurons are to be compared for each feature-layer. Based on the statistical interpretation of $\boldsymbol{W}^{i,i+1}$s and $\Phi_i$s we *choose* to compare neurons in the order of activation magnitude. So, for each $\Phi_i$, the neuron outputs are ordered by the activation magnitude in the descending order, then the distance proxy is computed between neurons with the same magnitude-rank. Figure 6 plots the result.

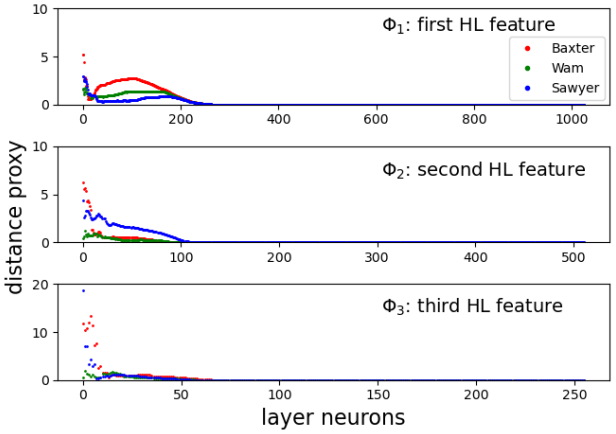


Fig. 6. Distance proxy giving the neuron-wise difference of the cumulative activation over the successive robot motion for $[a,b]_j, j \in (1,\dots,7)$ between Kuka and Baxter (red), WAM (green), and Sawyer (blue), for $\Phi_1$ (top), $\Phi_2$ (middle), and $\Phi_3$ (bottom).

An interesting characteristic we find is how close $\Phi_i$s are between each robot, indicated by the vast majority of neurons showing no difference between robots at each layer. Focusing on the difference seen for neurons with highest activation values, Baxter again shows the largest deviation from other robots at all layers. Further investigation of the underlying mechanisms of these differences should reveal additional insights which we will leave for future work.

## VI. Conclusion

In this work, a deep swept volume continuous function approximator, $\widetilde{\mathcal{SV}}$, was transfer-learned between multiple robots via multitask learning. We demonstrated that multitask learning is beneficial for extracting source information to be transferred. The source was the kinematic and geometric information common to the robots in the form of feature mappings. Using this source, positive transfer results were

obtained for learning a new robot's $\widetilde{\mathcal{SV}}$ where up to 5.9% predictor accuracy was shown. Moreover, the transfer learning paradigm allowed to learn the $\widetilde{\mathcal{SV}}$s with much smaller amount of training data, on the order of thousand-fold. This reduction is impactful on the computation time required for the training data generation. Additionally, we analyzed the DNNs' weights and feature mappings and provided insights into the transferability patters seen in the transfer experiments.

## REFERENCES


[1] T. Lozano-Perez, "Spatial planning: A configuration space approach," in *Auto. Robot Vehicles*, 1990, pp. 259–271.

[2] N. Makondo and B. Rosman, "Towards improving incremental learning of manipulator kinematics with inter-robot knowledge transfer," in *Proc. S. Africa Univ. Powe. Eng. Conf./Robot. and Mechatron./Patt. Recog. Ass. of S. African (SAUPEC/RobMech/PRASA)*, 2019, pp. 68–73.

[3] N. Makondo, B. Rosman, and O. Hasegawa, "Accelerating model learning with inter-robot knowledge transfer," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 2417–2424.

[4] B. Bócsi, L. Csató, and J. Peters, "Alignment-based transfer learning for robot models," in *Proc. Int. Joint. Conf. on Neur. Netw. (IJCNN)*, 2013, pp. 1–7.

[5] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on Knowledge and Data Engineering*, vol. 22, no. 10, pp. 1345–1359, Oct 2010.

[6] K. Weiss, T. M. Khoshgoftaar, and D. Wang, "A survey on transfer learning," *Journal of Big Data*, vol. 3, no. 9, 2016.

[7] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, "The limits and potentials of deep learning for robotics," *Int. J. Robot. Res.*, vol. 37, no. 4-5, pp. 405–420, 2018.

[8] K. Pereida, M. K. Helwa, and A. P. Schoellig, "Data-efficient multi-robot, multitask transfer learning for trajectory tracking," *Robot. and Automat. Lett.*, vol. 3, no. 2, pp. 1260–1267, 2018.

[9] H.-T. L. Chiang, A. Faust, S. Sugaya, and L. Tapia, "Fast swept volume estimation with deep learning," in *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, M. Morales, L. Tapia, G. Sanchez-Ante, and S. Hutchinson, Eds. Springer, 2018.

[10] H.-T. L. Chiang, J. E. Baxter, S. Sugaya, M. R. Yousefi, A. Faust, and L. Tapia, "Fast deep swept volume estimator," *Int. J. Robot. Res.*, 2020.

[11] J. Baxter, M. R. Yousefi, S. Sugaya, M. Morales, and L. Tapia, "Deep prediction of swept volume geometries: Robots and resolutions," in *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*, 2020, pp. 6665–6672.

[12] Y. J. Kim, G. Varadhan, M. C. Lin, and D. Manocha, "Fast swept volume approximation of complex polyhedral models," *Computer-Aided Design*, vol. 36, no. 11, pp. 1013–1027, 2004.

[13] S. Abrams and P. K. Allen, "Computing swept volumes," *The Journal of Visualization and Computer Animation*, vol. 11, no. 2, pp. 69–82, 2000.

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

[15] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical planning in the now," in *Proc. Int. Conf. Artif. Intel.*, 2010, pp. 33–42.

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

[17] J. J. Kuffner, "Effective sampling and distance metrics for 3D rigid body path planning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2004, pp. 3993–3998.

[18] M. Campen and L. Kobbelt, "Polygonal boundary evaluation of Minkowski sums and swept volumes," in *Computer Graphics Forum*, vol. 29, no. 5, 2010, pp. 1613–1622.

[19] R. Caruana, "Learning many related tasks at the same time with backpropagation," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. S. Touretzky, and T. K. Leen, Eds. MIT Press, 1995, pp. 657–664.

[20] ——, "Multitask learning," *Machine Learning*, vol. 28, no. 1, pp. 41–75, 1997.

[21] J. C. Himmelstein, E. Ferre, and J.-P. Laumond, "Swept volume approximation of polygon soups," *IEEE Trans. on Autom. Sci. and Eng.*, vol. 7, no. 1, pp. 177–183, 2010.

[22] A. Von Dziegielewski, M. Hemmer, and E. Schömer, "High precision conservative surface mesh generation for swept volumes," *IEEE Trans. on Autom. Sci. and Eng.*, vol. 12, no. 1, pp. 183–191, 2015.

[23] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251–257, 1991.

[24] M. E. Taylor and P. Stone, "Transfer learning for reinforcement learning domains: A survey," *Journal of Machine Learning Research*, vol. 10, no. 56, pp. 1633–1685, 2009.

[25] Z. Zhu, K. Lin, and J. Zhou, "Transfer Learning in Deep Reinforcement Learning: A Survey," *arXiv e-prints*, p. arXiv:2009.07888, Sept. 2020.

[26] M. Hazara and V. Kyrki, "Transferring generalizable motor primitives from simulation to real world," *Robot. and Automat. Lett.*, vol. 4, no. 2, pp. 2172–2179, 2019.

[27] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE Trans. Neural. Netw. Learn. Syst.*, vol. 26, no. 5, pp. 1019–1034, 2015.

[28] Z. Kira, "Inter-robot transfer learning for perceptual classification," in *Proc. Int. Conf. Auton. Agent. Multi. Agent. Syst.* Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems, 2010, p. 13–20.

[29] G. Costante, T. A. Ciarfuglia, P. Valigi, and E. Ricci, "Transferring knowledge across robots: A risk sensitive approach," *Robotics and Autonomous Systems*, vol. 65, pp. 1–14, 2015.

[30] T. Weng, A. Pallankize, Y. Tang, O. Kroemer, and D. Held, "Multi-modal transfer learning for grasping transparent and specular objects," *Robot. and Automat. Lett.*, vol. 5, no. 3, pp. 3791–3798, 2020.

[31] M. K. Helwa and A. P. Schoellig, "Multi-robot transfer learning: A dynamical system perspective," in *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*, 2017, pp. 4702–4708.

[32] P. Englert and M. Toussaint, "Kinematic morphing networks for manipulation skill transfer," in *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*, 2018, pp. 2517–2523.

[33] J. Stüber, M. Kopicki, and C. Zito, "Feature-based transfer learning for robotic push manipulation," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2018, pp. 5643–5650.

[34] S. Ruder, "An Overview of Multi-Task Learning in Deep Neural Networks," *arXiv e-prints*, p. arXiv:1706.05098, June 2017.

[35] M. Crawshaw, "Multi-Task Learning with Deep Neural Networks: A Survey," *arXiv e-prints*, p. arXiv:2009.09796, Sept. 2020.

[36] L. Pinto and A. Gupta, "Learning to push by grasping: Using multiple tasks for effective learning," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, 2017, pp. 2161–2168.

[37] N. Radwan, A. Valada, and W. Burgard, "Vlocnet++: Deep multitask learning for semantic visual localization and odometry," *Robot. and Automat. Lett.*, vol. 3, no. 4, pp. 4407–4414, 2018.

[38] S. Liu, E. Johns, and A. J. Davison, "End-to-End Multi-Task Learning with Attention," in *Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 1871–1880.

[39] G. Gkioxari, B. Hariharan, R. Girshick, and J. Malik, "R-CNNs for Pose Estimation and Action Detection," *arXiv e-prints*, p. arXiv:1406.5212, June 2014.

[40] R. Caruana and J. O'Sullivan, "Multitask pattern recognition for autonomous robots," in *Proc. IEEE Int. Conf. on Intel. Robot. Sys. (IROS)*, vol. 1, 1998, pp. 13–18.

[41] S. Choi and J. Kim, "Cross-domain motion transfer via safety-aware shared latent space modeling," *Robot. and Automat. Lett.*, vol. 5, no. 2, pp. 2634–2641, 2020.

[42] Z. Chen, V. Badrinarayanan, C.-Y. Lee, and A. Rabinovich, "GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks," in *Proc. of Int. Conf. on Machine Learning*, J. Dy and A. Krause, Eds., vol. 80. Stockholm, Sweeden: PMLR, 10–15 Jul 2018, pp. 794–803.

[43] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA: MIT Press, 2017.

[44] C. A. Pruneau, *Data Analysis Techiques for Physical Scientists*. New York, NY: Cambridge University Press, 2017.

[45] G. E. Shilov, *Elementary Functional Analysis*. New York, NY: Dover Publications, Inc., 1974.