https://doi.org/10.20965/jrm.2018.p0265

Paper:

A Non-Linear Manifold Alignment Approach to Robot Learning from Demonstrations

Ndivhuwo Makondo[*1,*2], **Michihisa Hiratsuka**[*3], **Benjamin Rosman**[*2,*4], **and Osamu Hasegawa**[*5]

[*1]Interdisciplinary Graduate School of Science and Engineering, Tokyo Institute of Technology
4259 Nagatsuta-cho, Midori-ku, Yokohama 226-8503, Japan
E-mail: ndivhuwo9@gmail.com
[*2]Mobile Intelligent Autonomous Systems, Council for Scientific and Industrial Research
Meiring Naude Road, Brummeria, Pretoria 0001, South Africa
[*3]Data Engineering Group, Recruit Lifestyle Co., Ltd.
1-9-2 Marunouchi, Chiyoda-ku 100-6640, Japan
E-mail: m.hiratsuka0716@gmail.com
[*4]School of Computer Science and Applied Mathematics, University of the Witwatersrand
Meiring Naude Road, Brummeria, Pretoria 0001, South Africa
E-mail: brosman@csir.co.za
[*5]Faculty of Engineering, Department of Systems and Control Engineering, Tokyo Institute of Technology
4259 Nagatsuta-cho, Midori-ku, Yokohama 226-8503, Japan
E-mail: oh@haselab.info

[Received July 3, 2017; accepted January 15, 2018]

The number and variety of robots active in real-world environments are growing, as well as the skills they are expected to acquire, and to this end we present an approach for non-robotics-expert users to be able to easily teach a skill to a robot with potentially different, but unknown, kinematics from humans. This paper proposes a method that enables robots with unknown kinematics to learn skills from demonstrations. Our proposed method requires a motion trajectory obtained from human demonstrations via a vision-based system, which is then projected onto a corresponding human skeletal model. The kinematics mapping between the robot and the human model is learned by employing Local Procrustes Analysis, a manifold alignment technique which enables the transfer of the demonstrated trajectory from the human model to the robot. Finally, the transferred trajectory is encoded onto a parameterized motion skill, using Dynamic Movement Primitives, allowing it to be generalized to different situations. Experiments in simulation on the PR2 and Meka robots show that our method is able to correctly imitate various skills demonstrated by a human, and an analysis of the transfer of the acquired skills between the two robots is provided.

Keywords: learning from demonstrations, knowledge transfer, multi-robot systems, manifold alignment

1. Introduction

The number of robots active in real-world settings is growing and their capabilities are continuously improving as well. In some instances these robots are beginning to assist humans with a variety of tasks in every day environments, such as cooking, washing dishes and delivering cutlery in kitchens; helping doctors with surgical procedures in hospitals; elderly care; grocery shopping, etc. Some of these tasks may be unknown prior to the commissioning of the robot. Therefore it is important that robots can be adapted to new situations by extending their sets of behaviors or skills.

Conventionally, robot skills – also known as policies or behaviors – are developed by hand, where they are hard-coded onto robots by an engineer. However, this requires tedious effort and expertise; and to adapt to new situations, new behaviors need to be hard-coded onto the robots. This process is not accessible to a wider range of non-robotics-expert users. Recently however, machine learning techniques have been adopted to enable robots to acquire policies from data. This offers opportunities for robots to continuously build new policies as new data arrives, as well as adapt to new situations. One such technique is robot Learning from Demonstrations (LfD).[1]

In LfD, a policy is learned from example data sets provided by a demonstrator. The demonstrator acts as a teacher, either in the form of a human or another robot, performing desired behaviors for the robot to learn. Without loss of generality we will refer to the teacher as a human, as is illustrated in **Fig. 1**. LfD algorithms utilize the provided data sets to derive policies that attempt to reproduce and generalize the desired demonstrated behaviors on the robot [1]. This is in contrast to other techniques in which robots learn from their own experiences, either through optimization of some reward function provided by a human, as in reinforcement learning [2]; or

1. LfD is sometimes known as Imitation Learning (IL), Programming by Demonstration (PbD) and Learning by Demonstration (LbD), however the term LfD has gained popularity recently.

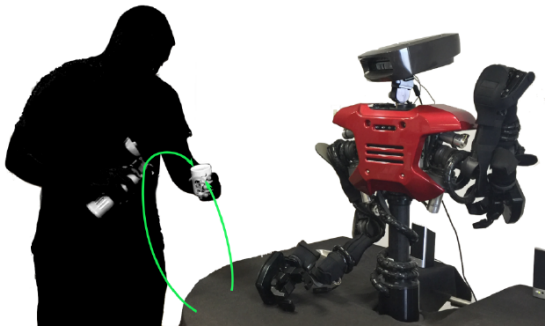


Fig. 1. Example of goal-directed imitation learning from human demonstrations. A robot learner uses a Kinect sensor to observe a human teacher demonstrate a task of pouring into a cup. The robot learner must be able to reproduce the general task of pouring into the cup, rather than merely mimicking the human posture. The curves represent trajectories of the objects during teacher demonstration.

through autonomous self-exploration, as in developmental robotics [3].

Within LfD, policy acquisition can be facilitated by ordinary users (i.e., non-robotics-expert users), because its formulations do not typically require domain expertise of the robots and tasks. Furthermore, demonstration is intuitive for humans as they already use it to teach other humans, making it natural to demonstrate tasks to robots. The LfD learning problem can be broadly segmented into two phases: how to gather demonstrations and how to derive policies [1]. Gathering demonstrations is the process of building a data set of examples, which ranges from the selection of sensors for collecting the data – which controls the type of data collected – to the type of demonstration technique to use. Deriving a policy generally involves encoding the provided data set of examples by learning a model, which can be used to later reproduce the demonstrated task on the robot and generalize to different contexts.

Within the context of gathering demonstrations, various techniques, or strategies, exist. A human teacher can provide demonstrations by tele-operating the robot learner with a joystick while performing the desired behavior [4], physically guiding the robot learner through the task (also known as kinesthetic guiding) [5], or recording the demonstrations using vision-based systems, such as a motion capture system, while the teacher performs the task herself [6].

Each strategy has its own limitations. In tele-operation and kinesthetic guiding, sensors that record the demonstration are typically placed on the robot, so the data sets collected can be directly used by the robot learner. However, these strategies are limited to robots with fewer degrees-of-freedom (DoF), due to the difficulty of a human controlling each DoF in order to produce a coordinated overall behavior. Strategies that record a human demonstrator using vision-based systems are natural for humans as they allow the human teacher to perform the task as best as they can, without any obstruction from the robot learner. However, this suffers from so-called correspondence issues [7] – issues that arise due to the differences in the kinematics of the teacher and robot learner, which prevent the collected data sets from being directly used by the robot learner.

With regards to deriving a policy from the provided data sets, data-driven approaches have received considerable attention recently. Statistical modeling [5, 8] and dynamical systems [9, 10] are amongst the most popular approaches. In particular, dynamic movement primitives (DMPs) are widely used due to their flexibility and stability [10–12], and have been used to parameterize policies for reinforcement learning of continuous robotic motions in high dimensions.

The focus of this paper is on how to adapt demonstrations gathered using vision-based systems, such that they are useful to the robot learner, and to demonstrate that policies can be derived from the adapted demonstrations. Several approaches have been proposed to deal with correspondence issues that arise in this setting, including optimization approaches based on a kinematic model of the robot learner [6, 13, 14], and data-driven approaches that attempt to model a mapping from the teacher space to the robot space [15–17]. Techniques that rely on a kinematic model of the robot learner are among the most widely used, and some general methods have been proposed [6, 14]. However, in some cases an accurate kinematic model of the robot learner may not be available, limiting the applicability of these methods.

This may be the case, for example, when the specifications required for modeling the robot are not released by the manufacturer [18], and so need to be measured by hand, leading to an inaccurate kinematic model, requiring further calibration. This may also be the case when working with robots whose bodies are not static, potentially due to modification, repair, or material damage [19]; or when dealing with biologically-inspired robots, with realistic skeletons and series-elastic, compliant actuators, such as Coman [20] and Meka [a]; and tendon-driven joints, such as the iCub [21]. In these cases, data-driven approaches based on machine learning techniques offer an alternative.

In this paper we propose a data-driven approach based on manifold alignment that is suitable in such cases where knowledge of the kinematic model of the robot learner is not available. This paper is an extension of our previous published work [22], in which we showcased the extraction of simple human movements using a Kinect sensor and transferring them to a single robot, where the workspaces of the human teacher and the robot were assumed to intersect. Here, we formalize the problem and greatly extend our analysis to two humanoids learning more complex tasks from demonstration, showcasing the robustness of our method to differences in kinematics, and a comparison of two manifold alignment techniques. To address the issue of non-intersecting workspaces, we propose an approach for aligning the workspaces based on rough alignment of some data generated in both

workspaces. Lastly, we provide an analysis of transferring skills that have been acquired by one robot from human demonstrations, to another robot. This analyzes the loss incurred over multiple rounds of transfer, compared to directly transferring again from a human teacher, who may not always be available.

The rest of the paper is organized as follows. Section 2 provides an overview of related work in human motion adaptation to kinematically different embodiments. In Section 3 a general description of the problem statement is given. Section 4 outlines our proposed method. In Section 5 the experimental setup is described, and the experimental results are presented. Finally, we conclude the paper in Section 6.

2. Related Work

The idea of projecting a motion from one kinematic embodiment to another is also found in other areas of computer graphics and robotics, where it is generally referred to as kinematic retargeting [14, 23]. It allows the transfer of gestures or behaviors that are defined in one reference frame (the source) to another (the target) [14]. Techniques generally differ based on the type of motions to be adapted and information to be preserved. For example, to adapt dancing motions from a human dancer to a humanoid, we may be interested in adapting the *joint configurations* of the human dancer, such that the robot can mimic the dance moves, thus preserving postural information. Another example of motions, which is of particular interest to our work, are motions in which we instead desire the preservation of *goal-directed* characteristics of the movement, where the focus is on achieving some goal, typically with an end-effector of the robot. We refer to this as goal-directed imitation.[2]

Example cases where goal-directed imitation is useful include a workshop setting, where it may be desired for multiple robots to perform some tasks demonstrated by a human, such as painting, welding, or pouring fluids as shown in **Fig. 1**. Another example is playing golf, where the robot must swing the golf club such that it strikes the golf ball at a desired location while satisfying some constraints such as via-points and obstacles [6]. For a robot to successfully reproduce such tasks, it must satisfy some constraints in task space, rather than merely mimicking the human movements. In our experiments, we use a goal-directed task of writing letters in the task space, where it is desired that the robot learner reproduce the letters exactly in size and position relative to the learner's reference frame.

Approaches based on kinematic retargeting for adapting this kind of motion to robots typically assume an accurate kinematic model of the target robot. The general idea is to find an optimal transformation (i.e., locating the task in the robot frame) and adaptation of the demonstration to the target robot, where correspondences between the human and the robot are typically known. The demonstrations are generally adapted by maximizing their similarity to the reproductions by the target robot, while satisfying kinematic constraints, such as joint limits and end-effector reach. This includes techniques based on non-linear optimization [24], using Inverse Kinematics (IK) to fit corresponding poses between the human body structure and the robot structure [25], and optimizing a generic weighted cost function whose weights control the similarity of tasks in both task and joint spaces [14]. When correspondences are not known (e.g., adapting to non-anthropomorphic robots), an automatic method that searches for the optimal location and adaptation of the human demonstration, based on the capability of the robot to reproduce it, can be used [6].

When the kinematic data for the target robot is not available, data-driven approaches have been employed as alternatives. Here, a data set of correspondences between the source (human or robot) and target (robot) spaces is collected and a mapping between the spaces is learned, after which this mapping is used to transfer novel points from the source to the target space. Most techniques in the literature transfer human demonstrations in joint space, typically without explicitly addressing goal-directed motions.

Examples include learning a direct mapping from sensor data from a motion capture suit to the position of the robot actuator by training a feed-forward neural network for each DoF [15]; or a two-step mapping process, where the sensor data and robot actuator data are assumed to share a common latent space of lower dimensionality, and the goal is to find mappings from the high-dimensional sensor data to the latent space and then to the high-dimensional robot actuator space. Several techniques have been proposed for learning these mappings, including treating the task as a regression problem and applying Gaussian processes for both mappings [17, 26], using Kernel Canonical Correlation Analysis (KCCA) to map to the latent space and Kernel Regression (KR) to map from the latent space to the robot space [27], and using a mixture of factor analyzers (MFA) combined with a dynamical system for modeling and stable reproduction of trajectories [28].

Another example is using Shared Gaussian Process Latent Variable Models (Shared-GPLVM) to jointly learn a latent representation of skills in a lower-dimensional space [16]. This has shown to be able to use the hyperparameters of one robot to accelerate learning of the same skills by another kinematically similar robot. This is similar to our work in that knowledge acquired by one robot from demonstrations is transferred to other robots, which reduces the time the human operator spends on training the robots. However, the Shared-GPLVM models as presented assume that the source and target inputs coincide, which is not necessarily the case if the robots do not share the same workspace as the human demonstrator. Furthermore, goal-directed motions were not addressed in this work.

2. The term goal-directed imitation is also used to mean imitation learning where the teacher's intention is inferred from the demonstrations. Here we use it to describe imitation of whole trajectories in task space.

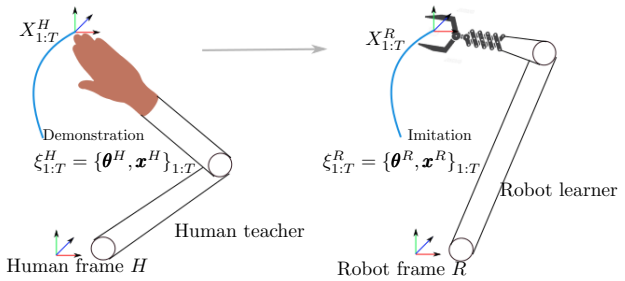


Fig. 2. Adapting human demonstrations onto robot learner.

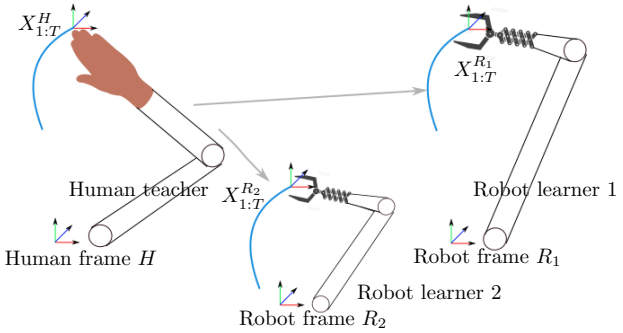


Fig. 3. Multi-robot problem setting for two robot learners.

Finally, in our previous work [22] we proposed learning a mapping directly from sensor data (projected onto a human skeletal model) to robot actuator space, using Local Procrustes Analysis (LPA). However, we required that the human teacher and the robot share the same workspace, and we only considered the case of teaching a single robot. In this paper we relax the requirement of shared workspaces, by allowing a transformation of the demonstrator's task space into the robot learner frame, such that their workspaces overlap and the demonstrations can be performed by the robot (similar to kinematic retargeting). We also provide more extensive experimental results on two humanoids performing more complex tasks, showcasing the robustness of our method to differences in kinematics, a comparison of LPA to a baseline linear Procrustes Analysis, and an analysis of transfer of skills acquired from human demonstrations between the robots, in which we determine the loss incurred over multiple rounds of transfer compared to transferring directly to the second robot from a human teacher.

3. Problem Statement

Assume a given trajectory $\xi^H_{1:T} = \{\boldsymbol{\theta}^H, \boldsymbol{x}^H\}_{1:T}$ of duration T, provided by a human demonstrator H, consisting at each time step of a d_H dimensional human joint angle vector $\boldsymbol{\theta}^H$ and a human hand (tip, end-effector, etc.) position and orientation (pose) vector $\boldsymbol{x}^H$ in task space X^H, where d_H is the number of the human joints active when performing a particular task. $\boldsymbol{x}^H$ is given in the reference frame of the human demonstrator. A robot R in a different location is to learn a parameterized skill policy $\pi_{\boldsymbol{\beta}}$, that reproduces a generalized form of the demonstrated trajectory with its own arm w.r.t. to its own reference frame. $\boldsymbol{\beta}$ is a vector of the policy parameters that must be learned from the robot data.

We aim to adapt $\xi^H_{1:T}$ to $\xi^R_{1:T} = \{\boldsymbol{\theta}^R, \boldsymbol{x}^R\}_{1:T}$ such that the robot is able to reproduce it, where $\boldsymbol{\theta}^R$ is a d_R dimensional joint angle vector of the robot, consisting of joints active when the robot is performing the task and $\boldsymbol{x}^R$ is the pose vector of the robot end-effector in robot Cartesian space X^R. This is illustrated in **Fig. 2**. The adapted trajectory data $\xi^R_{1:T}$ is then used to learn the policy parameters $\boldsymbol{\beta}$.

We aim to adapt the given human demonstration without assuming a kinematic model of the target robot, in contrast to much of the related work. This can be extended into a multi-robot problem setting (as illustrated by **Fig. 3** for $n = 2$ robot learners), where the demonstration $\xi^H_{1:T}$ must be adapted to multiple robot trajectories $\xi^{R_1}_{1:T}, \xi^{R_2}_{1:T}, \ldots, \xi^{R_n}_{1:T}$, for n robots. Each trajectory is encoded onto its own skill policy for the corresponding robot. Thus, n different mappings must be learned from the human data to the n robots. In the next section we present our proposed data-driven approach for solving this problem.

4. Proposed Method

Our proposed method employs a data-driven scheme based on manifold alignment, that maps data from the domain of one agent to another. It requires that we provide corresponding samples from the domains, and a nonlinear mapping is learned from these samples. This mapping must generalize to samples from the same domains not seen during training. The domains represent kinematic data generated by a human or robot.

An alternative approach would be to instead estimate kinematic parameters of the robot learner (e.g., link lengths) or the inverse kinematics model from the sampled robot data, and then employ kinematic retargetting. Although this approach is interesting, learning kinematics models accurately, typically framed as a regression problem, often requires a large amount of data, which is particularly costly in the case of robots [29, 30].

In [29], it has been shown that a mapping between kinematic domains can be learned from very few samples using Local Procrustes Analysis, compared to the number of samples required to instead learn a kinematic model of a robot. The learned mapping could be further used to provide more samples for learning a kinematics model of a target robot, by mapping samples generated by another robot. This is an instance of a general transfer learning problem, in which estimating a mapping between different domains, from a few samples of correspondences, is typically easier than collecting a sufficient amount of data in the target domain to accurately learn a

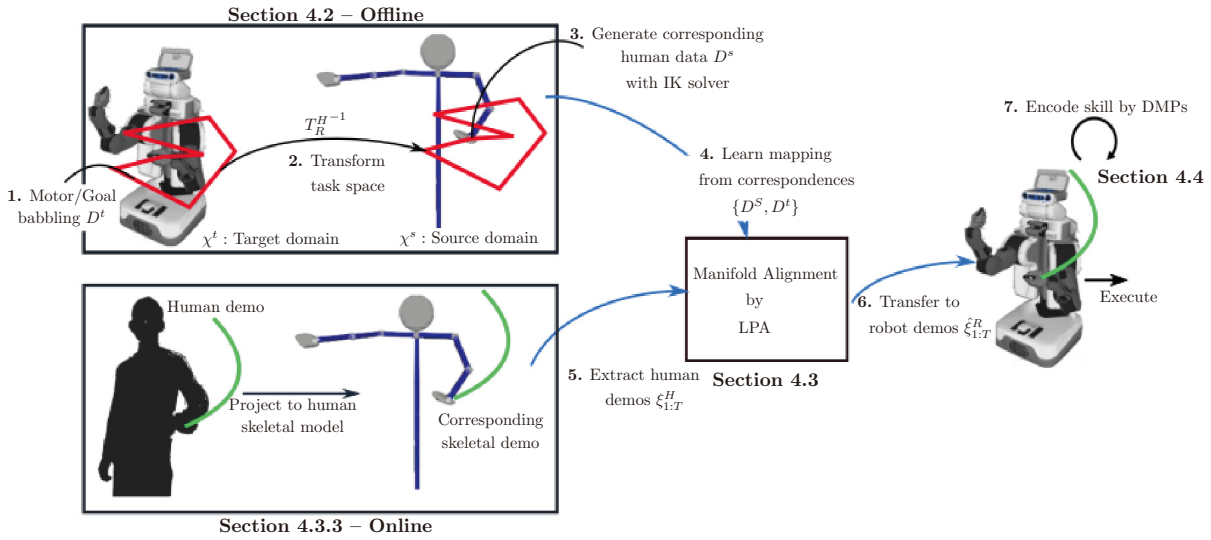


Fig. 4. Overview of proposed method.

target model [29–31].

Below, a high-level overview of our method is presented (Section 4.1), followed by the processes of collecting sample correspondences between the domains (Section 4.2), learning the mapping (Section 4.3), and encoding the mapped trajectories as parametrized skills (Section 4.4).

4.1. Overview

Figure 4 shows the overview of our proposed method. Given human demonstrations, represented as trajectories of a human skeletal model in joint space and corresponding points in task space, we aim to map the trajectories onto the body of a robot, such that we obtain joint-space trajectories for the robot corresponding to the human demonstrations. These mapped trajectories are subsequently encoded as parametrized skills for later use in new situations. We adopt the approach of projecting human captured data onto a corresponding human skeletal model, because this allows for a unified representation of captured human data from different sensors [32–35], and also enables our method to be applied in conjunction with different motion capture systems.

In the first phase of our method, corresponding samples between the human (source) and robot (target) spaces are collected. These samples must be representative of the respective spaces in which the human teacher demonstrates the tasks and the robot learner is expected to perform the tasks. This phase is composed of steps 1–3 in **Fig. 4** and is described in Section 4.2. Then a non-linear mapping is learned from these samples using LPA in step 4 of **Fig. 4**, as described in Section 4.3.

The second phase is composed of steps 5–7 in **Fig. 4**, where the learned mapping is employed to adapt human joint trajectories onto robot joint trajectories, as described in Section 4.3.3, after which the adapted trajectories are encoded as parametrized skills (see Section 4.4). Since the learned mapping is defined globally within the limits of the given spaces, any trajectory that lies in the domain of the human space can be mapped onto the robot space. This consequently allows the transfer of any skills that both the human and the robot are capable of performing within their respective domains.

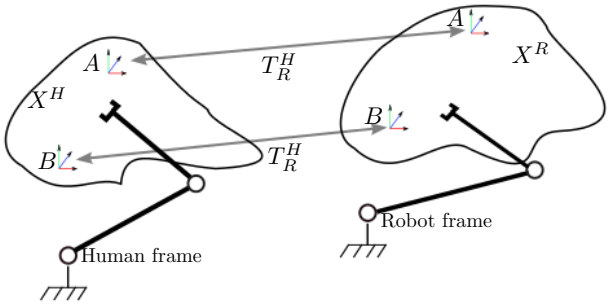
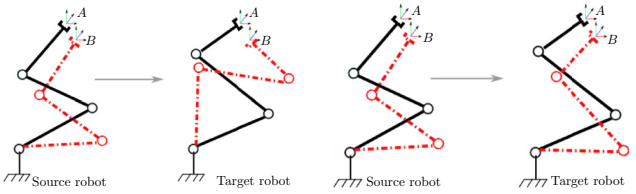


Fig. 5. Illustration of workspace alignment T_R^H between a human model and robot with different reference frames.

4.2. Correspondence

As described in Section 2, goal-directed imitation requires that the robot preserves task-space information. For this reason, we define correspondences in end-effector space. That is, a human joint configuration and a robot joint configuration correspond if they both reach the same end-effector pose defined in the same reference frame. In the general case, in which the end-effector spaces of two arms do not overlap, a workspace alignment T_R^H is needed to be able to relate points in the two end-effector spaces. This is illustrated in **Fig. 5**.

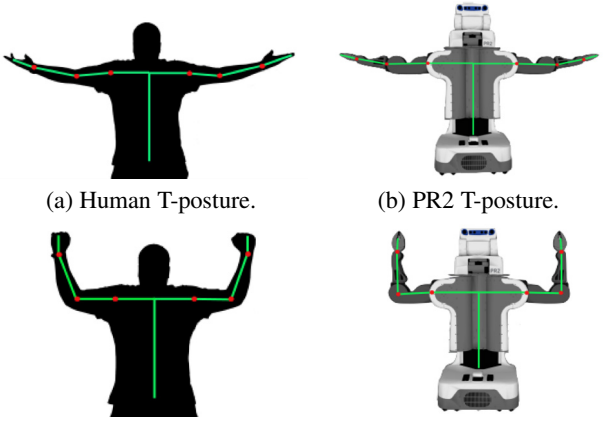
For redundant arms (e.g., a 7-DoF human or robot arm in a 3D space) there may be an infinite number of joint configurations that reach the same end-effector pose. Thus, using correspondences in the end-effector space alone can cause configurations in one joint space to cor-



(a) Inconsistent IK solutions for neighboring points A and B due to redundancies.

(b) Consistent IK solutions for neighboring points A and B with redundancy resolution.

Fig. 6. Illustration of inconsistent IK solutions due to arm redundancies for 3-DoF planar robots in a 2D task space. The solid and dashed lines correspond to similar tasks for the same robot. (a) Target data set will have inconsistent neighbors due to redundancies. (b) With redundancy resolution, target data set will have consistent neighbors.



(a) Human T-posture.

(b) PR2 T-posture.

(c) Human U-posture.

(d) PR2 U-posture.

Fig. 7. Illustration of the T-posture and U-posture with the PR2 robot.

respond to configurations potentially lying far from each other in another joint space, resulting in an inconsistent sample of correspondences, as illustrated in **Fig. 6(a)**. To resolve this inconsistency, in addition to joint configurations reaching the same end-effector pose, we require that they also assume similar postures, resulting in consistent configurations shown in **Fig. 6(b)**.

The similarity of postures can be determined visually by the human teacher [15, 16]. To this end, the robot is programmed to assume a pre-defined sequence of postures, such as the T-posture [36] and U-posture shown in **Fig. 7**. Then the human teacher mimics these robot postures, thus generating corresponding human postures. From this generated set of human-robot posture pairs, the teacher visually identifies the order of the robot joints and their movement in relation to hers (i.e., direction of rotation), and can thus generate a mapping f_{post} from robot postures to human postures, and vice versa. Given target robot configurations that reach A and B in **Fig. 6(b)**, we can use the posture mapping to select similar source robot postures $\boldsymbol{\theta}^s_{similar} = f_{post}\boldsymbol{\theta}^t$ out of the many that can reach A and B, thereby resolving redundancies consistently.

To collect a data set of correspondences, we propose

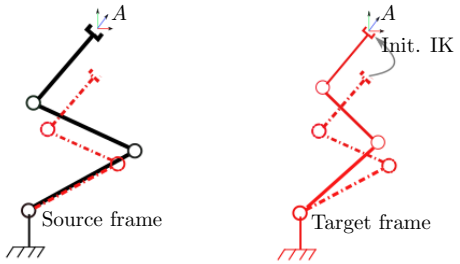


Fig. 8. Biasing IK for target arm with target posture similar to corresponding source arm posture. The dashed links represent the target arm posture similar to the source arm posture that reaches A (solid links in Source frame), and this is used to bias IK to find the target posture (solid in Target frame) that reaches A with a similar posture to that of the source arm.

generating N random robot pose data D^t, which can generally be accomplished by employing autonomous robot exploration strategies from developmental robotics, such as motor babbling or goal babbling [3, 37] – step 1 in **Fig. 4**. This data consists of robot joint angles $\boldsymbol{\theta}^t_i$ and their corresponding task space points $\boldsymbol{x}^t_i$, where $i = 1, 2, \ldots, N$ – i.e., $D^t = \{\boldsymbol{\theta}^t_i, \boldsymbol{x}^t_i\}$. Then we use a numerical IK solver on the human skeletal model to generate corresponding human joint angles $\boldsymbol{\theta}^s_i$, from their corresponding end-effector points[3] $\boldsymbol{x}^s_i = T_R^{H\,-1} \times \boldsymbol{x}^t_i$, forming the corresponding human data $D^s = \{\boldsymbol{\theta}^s_i, \boldsymbol{x}^s_i\}$ – steps 2–3 in **Fig. 4**.

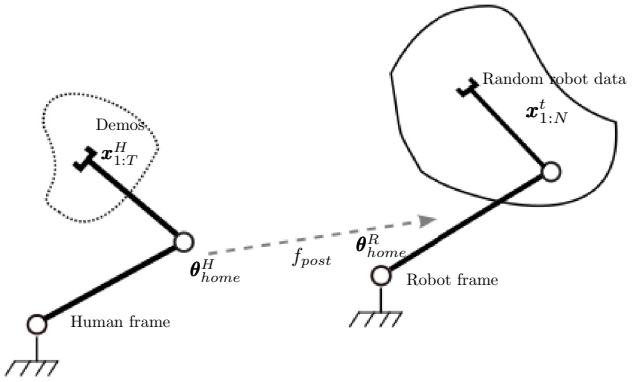
To avoid the inconsistencies that can arise due to arm redundancies, the IK solver is biased towards a solution consistent with the corresponding robot posture, by initializing it with a similar human posture generated using f_{post}, as illustrated in **Fig. 8**.

The workspace alignment T_R^H transforms points between the robot task space X^R and the human task space X^H, as shown in **Fig. 5**, and can be determined as an affine transformation as illustrated in **Fig. 9**. Firstly, we select a preferred human *home posture* $\boldsymbol{\theta}^H_{home}$ (e.g., at the center of the demonstrations $\boldsymbol{\theta}^H_{1:T}$) and map it to the robot joint space using the posture mapping f_{post}, to obtain a corresponding robot *home posture* $\boldsymbol{\theta}^R_{home}$. Then the N random robot poses $D^t = \{\boldsymbol{\theta}^t_i, \boldsymbol{x}^t_i\}$ are generated around $\boldsymbol{\theta}^R_{home}$. This is illustrated in **Fig. 9(a)**.

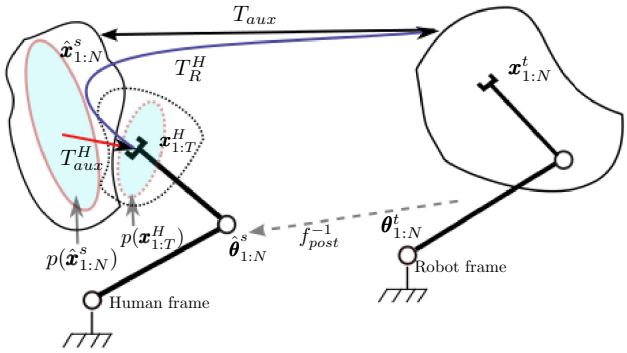
Secondly, as illustrated in **Fig. 9(b)**, we map the N random robot joints $\boldsymbol{\theta}^t_{1:N}$ into the human joint space, using the inverse of f_{post} (shown as the grey dashed arrow), effectively locating $\boldsymbol{x}^t_{1:N}$ in the human frame as *auxiliary* points $\hat{\boldsymbol{x}}^s_{1:N}$ using forward kinematics on the human model. Due to differences in kinematics, $\hat{\boldsymbol{x}}^s_{1:N}$ may not be exactly aligned with the human demonstrations $\boldsymbol{x}^H_{1:T}$, but this allows us to compute an auxiliary alignment T_{aux} (shown as the black solid arrow) from $\boldsymbol{x}^t_{1:N}$ to corresponding points $\hat{\boldsymbol{x}}^s_{1:N}$ in the human frame.

Then, thirdly, we can directly compare the human demonstrations $\boldsymbol{x}^H_{1:T}$ and the mapped auxiliary robot data

3. The inverse $T_R^{H\,-1}$ of the workspace alignment T_R^H maps points from the robot end-effector space to the human end-effector space.



(a) Generating random robot data near θ^R_{home}.



(b) Steps for computing the alignment T^H_R.

Fig. 9. Proposed workspace alignment approach. See text for details.

$\hat{\boldsymbol{x}}^s_{1:N}$ in the human frame. Since we do not have pairwise correspondences between $\boldsymbol{x}^H_{1:T}$ and $\hat{\boldsymbol{x}}^s_{1:N}$, we propose employing the *rough alignment* algorithm [30], in which we model $\boldsymbol{x}^H_{1:T}$ and $\hat{\boldsymbol{x}}^s_{1:N}$ as Gaussian distributed and find the alignment T^H_{aux} (shown as the short solid arrow between the two Gaussian ellipses) between them by minimizing the Kullback-Leibler divergence (KL) between the Gaussians $p(\boldsymbol{x}^H_{1:T})$ and $p(\hat{\boldsymbol{x}}^s_{1:N})$. The final alignment T^H_R is then computed as a composition of T^H_{aux} and T_{aux} (the curved solid arrow).

The KL-divergence between two Gaussian distributed data sets M^s and M^t has the following analytical form

$$2KL(p(M^s)||p(M^t)) = (\boldsymbol{\mu}_s - \boldsymbol{\mu}_t)^\top \Sigma_{tt}^{-1}(\boldsymbol{\mu}_s - \boldsymbol{\mu}_t) + tr(\Sigma_{ss}\Sigma_{tt}^{-1} - I) - ln|\Sigma_{ss}\Sigma_{tt}^{-1}|, \quad (1)$$

where $\boldsymbol{\mu}_s$ and $\boldsymbol{\mu}_t$ are the means of M^s and M^t, and Σ_{ss} and Σ_{tt} their covariance matrices. In the case where we only care about the translation between M^s and M^t we can model the linear alignment T^H_{aux} as the difference between their Gaussian centers $\boldsymbol{\mu}_s$ and $\boldsymbol{\mu}_t$; however, a general solution was derived in [30], using Eq. (1) as a cost function, and is as follows. First, the data sets are standardized by substracting the mean and whitening, and then obtaining the rotation A as follows

$$A = U_t \Lambda_t^{\frac{1}{2}} \Lambda_s^{-\frac{1}{2}} U_s^\top, \quad \ldots \quad (2)$$

where $U_s\Lambda_s U_s^\top$ is an eigenvalue decomposition of Σ_{ss} and similarly for $U_t\Lambda_t U_t^\top$ and Σ_{tt}. Thus, to map a random robot point $\boldsymbol{x}^t_i$ to its corresponding point $\boldsymbol{x}^s_i$ in the human frame, we first map it to its auxiliary point $\hat{\boldsymbol{x}}^s_i$ using T_{aux} and finally map it using T^H_{aux}, after it has been standardized[4] and then rotated by A; and we set $M^s = \{\hat{\boldsymbol{x}}^s_{1:N}\}$ and $M^t = \{\boldsymbol{x}^H_{1:T}\}$ to compute A in Eq. (2).

4.3. Learning the Mapping: Local Procrustes Analysis

Once a sample of corresponding points has been collected using the method presented in the previous section, we can use it to learn a mapping between two domains – step 4 in **Fig. 4**. To this end, we employ Local Procrustes Analysis [29]. Consider a scenario where there are two different but related domains: the source domain $\chi^s \subset \mathrm{R}^d$ and the target domain $\chi^t \subset \mathrm{R}^d$, where d is the dimensionality of the domains. In our case they belong to the source (human or robot) and the target robot respectively. They are related in the sense that their data is generated by kinematic chains,[5] and they are different due to the chains having different parameters (link lengths).

Given samples of correspondences from each domain, $D^s = \{\boldsymbol{\theta}^s_i, \boldsymbol{x}^s_i\}_{i=1}^N$ and $D^t = \{\boldsymbol{\theta}^t_i, \boldsymbol{x}^t_i\}_{i=1}^N$, where N is the sample size, the objective is to learn a mapping function, $f : \chi^s \mapsto \chi^t$, that maps data points from the source domain to the target domain, through which knowledge can be shared between the domains.

Manifold alignment techniques are useful in this kind of problem because they allow for knowledge transfer between two seemingly disparate data sets, by aligning their underlying manifolds [38, 39]. Applications include automatic machine translation [40], cross-lingual information retrieval [38, 39], transfer learning for Markov Decision Processes [38] and robot model learning [29, 30], object pose alignment [41, 42] and bioinformatics [38, 39, 42].

In knowledge transfer for robots, the aim is typically to avoid *expensive* data collection on the target robot. A linear manifold alignment algorithm based on Procrustes Analysis [38] was employed to show the possibility of learning a mapping between robots for accelerating learning of forward kinematics of a new robot [30], based on a small sample of correspondences generated from the robots. Procrustes Analysis represents the source and target data as manifolds, and assumes that the target manifold was generated by linearly transforming the source manifold. This allows determination of this linear transformation from a few samples of corresponding points between the manifolds, making it data efficient.

However, it has been shown that to align manifolds of robot kinematics data, non-linear functions are generally more accurate [29]. LPA extends the Procrustes Analysis algorithm to handle such non-linear mappings, while attempting to retain its data efficiency, by approximating a global non-linear manifold alignment with locally linear

4. The procedure is similar to the Procrustes Analysis without correspondences, discussed in Section 4.3.2.
5. A human (or robot) body can be described mathematically as a group of links (or rigid bodies) connected by joints, forming a kinematic chain.

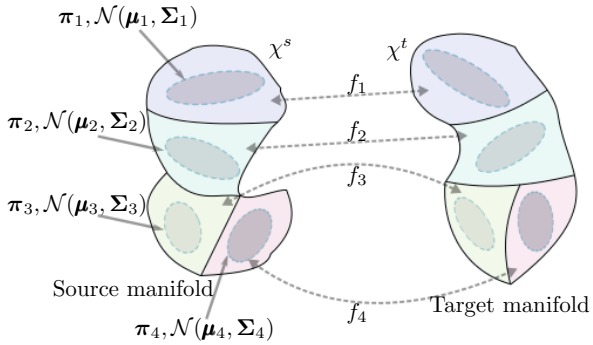


Fig. 10. Illustration of LPA with 4 clusters. The gray shaded ellipses represent GMM components and the shaded areas on the manifolds represent areas for which the GMM components are responsible. f_1 to f_4 are linear mappings learned from the data in their corresponding clusters.

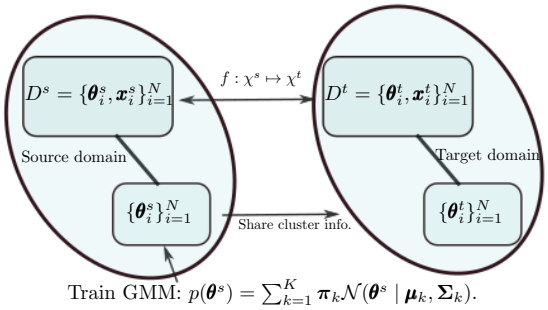


Fig. 11. Illustration of clustering in joint space. A GMM model is trained in the joint space of the source domain, and the clusters found are transferred to the joint space of the target domain, using the correspondence information. The non-linear mapping f is learned in all dimensions.

functions. This idea of approximating a global non-linear function with locally linear models has also been applied in locally weighted learning for control [43, 44] and learning non-linear image manifolds [45].

LPA assumes that the domains are locally continuous and smooth, and that the mapping can be computed locally using linear models on the corresponding instances $\boldsymbol{d}_i^s \leftrightarrow \boldsymbol{d}_i^t$, where $\boldsymbol{d}_i^s \in D^s$ and $\boldsymbol{d}_i^t \in D^t$. To achieve this, LPA first clusters the two data sets into K local clusters (see Section 4.3.1 and **Fig. 10**). Then a linear mapping for each cluster is computed using the Procrustes Analysis algorithm (see Section 4.3.2). A new data point from the source domain can then be mapped to the target domain by a weighted sum of the linear mappings (see Section 4.3.3).

4.3.1. Clustering and Mapping

This section describes how the two data sets can be clustered such that the weighted sum of the linear mappings, learned on the resulting clusters, yields a good non-linear mapping from the source domain to the target domain. The aim is to represent the two data sets D^s and D^t by a mixture of K regions, where corresponding points in the data sets map to the same region, as illustrated in **Fig. 10**.

LPA employs Gaussian Mixture Modeling (GMM), where the Gaussian mixtures correspond to the local regions, trained using the Expectation-Maximization (EM) algorithm, because it allows interpolating the output of local mappings using component responsibilities as weights. A GMM is represented by three parameters: the mixing coefficients π_k, the mean vectors $\boldsymbol{\mu}_k$ and the covariance matrices Σ_k. The total probability density over a vector $\boldsymbol{y}$ is then defined as a superposition of K Gaussian densities of the form

$$p(\boldsymbol{y}) = \sum_{k=1}^{K} \pi_k \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_k, \Sigma_k), \quad \ldots\ldots\ldots \quad (3)$$

and the components' responsibilities are defined as

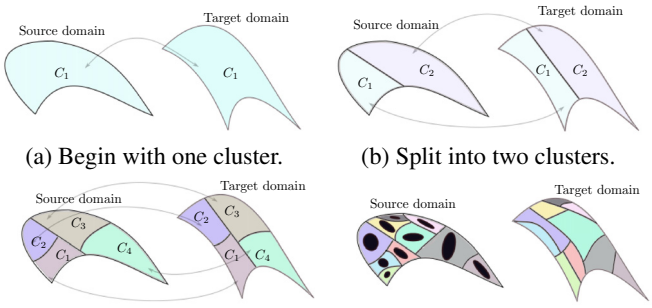
$$\gamma_k = \frac{\pi_k \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{y} \mid \boldsymbol{\mu}_j, \Sigma_j)}, \quad \ldots\ldots\ldots \quad (4)$$

where $\mathcal{N}$ is a multivariate normal distribution. γ_k can be viewed as the responsibility that component k takes for explaining the point $\boldsymbol{y}$.

In LPA, a GMM is trained on only one of the domains (the source domain in our experiments) and the data is clustered by assigning points to components with the highest responsibilities. This is indicated in **Fig. 10** by having GMM parameters only in the source domain. This clustering information, together with the information about correspondences, is then used to fit another GMM to the target domain, i.e., points in the target domain that correspond to points in the same cluster in the source domain are clustered together. Furthermore, our domain data consists of robot joints and end-effector positions, which are correlated through the kinematics of the body (i.e., moving arm joints affects the end-effector movement). So in order to efficiently obtain clusters, the GMMs are trained in the joint spaces of the data sets, as illustrated in **Fig. 11**.

In order to determine the number of GMM components K and initialize the parameters $\Pi = \{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$ for the EM algorithm, an algorithm was proposed in [29]. As there are many ways to initialize the EM algorithm, no restriction is put on this choice of initialization. The algorithm proposed in [29] employs a decisive hierarchical scheme, that automatically estimates the number of components (see **Algorithm 1** for the pseudocode). **Fig. 12** illustrates this procedure. It begins by training a GMM with one component and learning a linear mapping on its cluster, and then evaluating this mapping on the training data (**Fig. 12(a)**).

If the mapping error c_k is evaluated to be less than some pre-defined threshold $c_{\min}$ then the EM algorithm is initialized with one component and the corresponding parameters are initialized accordingly; otherwise the GMM



(a) Begin with one cluster. (b) Split into two clusters.

(c) Four clusters. (d) Final initialization.

Fig. 12. Illustration of the EM initialization procedure. In the final stage a GMM is learned in the source domain.

Algorithm 1 initEM

IN: Training sets D^s, D^t, $c_{\min}$, $N_{\min}$
Set cluster assignment vector $\boldsymbol{h}$ to ones and $K = 1$
while not terminated **do**
  for each GMM component $k \in [1, K]$ **do**
    Compute c_k on data belonging to its corresponding cluster C_k (see Eqs. (5) and (10))
    if $c_k > c_{\min}$ **then**
      Split cluster C_k into two ($C_{k,1}$ and $C_{k,2}$)
      if $N_{C_{k,1}} \geq N_{\min}$ and $N_{C_{k,2}} \geq N_{\min}$ **then**
        Update assignment vector $\boldsymbol{h}$ accordingly
      end if
    end if
  end for
  Update number of components K
end while
Compute each component parameters $\{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$
OUT: K, $\Pi = \{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$

component is split into two using the K-means algorithm and the whole process is repeated on the corresponding two new clusters (**Figs. 12(b)–(c)**). c_k is defined as follows

$$c_k = \frac{1}{N_k} \sqrt{\sum_{n=1}^{N_k} \| f_k(\boldsymbol{z}^s_{ik}) - \boldsymbol{z}^t_{ik} \|^2}, \quad \ldots\ldots (5)$$

where N_k is the number of points in cluster k, f_k is a linear mapping for cluster k (see Section 4.3.2), and $\boldsymbol{z}^s_{ik}$ and $\boldsymbol{z}^t_{ik}$ are corresponding data points in cluster k. This process is repeated until either (a) all c_k are less than the threshold $c_{\min}$, or (b) none of the clusters can be split further because one or both of the resulting clusters have number of points less than some pre-defined threshold $N_{\min}$.

The output of this procedure is the number of GMM components and the parameters $\Pi = \{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$ (**Fig. 12(d)**). In practice, this initialization procedure is run multiple times and parameters with the highest log-likelihood estimate are used to initialize the EM algorithm, which is run until convergence. After the EM algorithm has converged, clusters are created locally by assigning points to components with the highest responsibilities.

4.3.2. Linear Mapping with Procrustes Analysis

Given an assignment of points to local clusters as discussed in Section 4.3.1, we must learn a linear mapping for each cluster. To this end, we employ the Procrustes Analysis algorithm [30, 38]. The Procrustes Analysis is similar to the rough alignment algorithm discussed in Section 4.2, however Procrustes Analysis computes an *exact* alignment between two data sets based on provided pairwise correspondences. The goal is to find an optimal alignment from some source data set $Z^s \subset \mathrm{R}^d$ to some target data set $Z^t \subset \mathrm{R}^d$, where both data sets are assumed to have the same dimensionality d.

In our case Z^s and Z^t are data sets belonging to corresponding clusters in both source and target domains. Through this linear transformation, novel points in the source domain can be mapped onto the target domain. The optimal alignment is achieved by removing the translational, rotational and scaling components from one data set such that the two data sets are optimally aligned [38].

The first step in applying PA is to preprocess the data by subtracting the mean and whitening it, thus obtaining standardized matrices M^s and M^t, as follows:

$$\boldsymbol{s} = B^s(\boldsymbol{z}^s - \boldsymbol{\omega}^s), \quad \ldots\ldots (6)$$

$$\boldsymbol{t} = B^t(\boldsymbol{z}^t - \boldsymbol{\omega}^t), \quad \ldots\ldots (7)$$

where $\boldsymbol{s} \in M^s$ and $\boldsymbol{t} \in M^t$. The values $\boldsymbol{\omega}^s = \mathrm{E}\{Z^s\}$ and $\boldsymbol{\omega}^t = \mathrm{E}\{Z^t\}$ are the means of the data, where $\mathrm{E}\{\cdot\}$ denotes the expectation operator. Matrices B^s and B^t can be obtained using the Singular Value Decomposition (SVD) of the covariance matrices of Z^s and Z^t respectively, and are such that the data M^s and M^t are whitened. For example, the SVD decomposition of covariance Σ_z of matrix Z is $\Sigma_z = B\Lambda B^\top$, where B is the eigenvector and Λ a diagonal matrix of eigenvalues of Σ_z, ordered in decreasing order of eigenvalues. Here, B^s and B^t are eigenvectors obtained from SVD decompositions of covariance matrices of Z^s and Z^t respectively.

The manifold alignment function is modeled as a linear mapping $f_k : M^s \mapsto M^t$, with

$$f_k(\boldsymbol{s}) = A\boldsymbol{s} \quad \ldots\ldots (8)$$

where $A^{d\times d}$ is a transformation matrix. The expression for A was derived in [30], and is as follows:

$$A = \Sigma_{ss}^{-1}\Sigma_{ts}, \quad \ldots\ldots (9)$$

where Σ_{ss} is the covariance matrix of the source matrix M^s and Σ_{ts} is the covariance between the source and target matrices M^s and M^t. The reader is referred to [30] for a full derivation.

A new point $\boldsymbol{s}_\star = B^s(\boldsymbol{z}^s_\star - \boldsymbol{\omega}^s)$ in the source manifold can then be mapped using $\hat{\boldsymbol{z}}^t_\star = {B^t}^{\#} A\boldsymbol{s}_\star + \boldsymbol{\omega}^t$, where $\hat{\boldsymbol{z}}^t_\star$ is the transferred point and $B^\#$ is a Moore-Penrose inverse of B. **Algorithms 2** and **3** show pseudocode for learning and transferring using Procrustes Analysis.

Algorithm 2 Procrustes Analysis: Learning

1: IN: Cluster C_k training sets Z^s, Z^t
2: Compute M^s and M^t (see Eqs. (6) and (7))
3: Compute alignment matrix: $A = \Sigma_{ss}^{-1}\Sigma_{ts}$
4: OUT: Cluster C_k parameters $\Theta = \{A_k, B_k^s, B_k^t, \boldsymbol{\omega}_k^s, \boldsymbol{\omega}_k^t\}$

Algorithm 3 Procrustes Analysis: Transfer

1: IN: Cluster C_k parameters $\Theta = \{A_k, B_k^s, B_k^t, \boldsymbol{\omega}_k^s, \boldsymbol{\omega}_k^t\}$
2: IN: Source domain novel point $\boldsymbol{z}_\star^s$
3: Compute standardized novel point: $\boldsymbol{s}_\star = B_k^s(\boldsymbol{z}_\star^s - \boldsymbol{\omega}_k^s)$
4: Compute estimated target point: $\hat{\boldsymbol{z}}_\star^t = B_k^{t\#} A_k \boldsymbol{s}_\star + \boldsymbol{\omega}_k^t$
5: OUT: $\hat{\boldsymbol{z}}_\star^t$

Algorithm 4 Local Procrustes Analysis

1: IN: Training set D^s and D^t
2: IN: Parameters $c_{\min}$ and $N_{\min}$
3: $(\Pi,\ \mathrm{K}) \Leftarrow \mathrm{initEM}(D^s,\ D^t,\ c_{\min},\ N_{\min})$ {see **Algorithm 1**}
4: $\Pi \Leftarrow \mathrm{fitGMM}(\{\boldsymbol{\theta}_i^s\}_{i=1}^N, \Pi, \mathrm{K})$
5: **for** each cluster $k \in [1, K]$ **do**
6: Compute $\{A_k, B_k^s, B_k^t, \boldsymbol{\omega}_k^s, \boldsymbol{\omega}_k^t\}$ {see **Algorithm 2**}
7: **end for**
8: $\Theta = \{A_k, B_k^s, B_k^t, \boldsymbol{\omega}_k^s, \boldsymbol{\omega}_k^t\}_{k=1}^K$
9: OUT: $\{\Pi, \Theta\}$

Given this expression for the linear mapping, the mapping f_k in Eq. (5) can be substituted by this expression, obtaining the following expression for mapping error c_k:

$$c_k = \frac{1}{N_k}\sqrt{\sum_{n=1}^{N_k} \|B_k^{t\#} A_k \boldsymbol{s}_{ik} + \boldsymbol{\omega}_k^t - \hat{\boldsymbol{z}}_{ik}^t\|^2}. \quad \ldots (10)$$

4.3.3. Transfer with LPA

Once the GMM parameters, $\Pi = \{\pi_k, \boldsymbol{\mu}_k, \Sigma_k\}$, and corresponding linear mappings, $\Theta = \{A_k, B_k^s, B_k^t, \boldsymbol{\omega}_k^s, \boldsymbol{\omega}_k^t\}$, have been learned, we can build an LPA model and use it to transfer novel points from the source domain to the target domain. Note that only one GMM (in the source domain in our experiments) is required for interpolation when approximating the non-linear mapping, and that it is learned in the joint space, i.e., $\boldsymbol{\mu}_k$ is a vector in joint space and Σ_k is the covariance matrix of the joint angles. **Algorithm 4** summarizes the steps followed when learning an LPA model.

For a novel point $\boldsymbol{d}_\star^s = \{\boldsymbol{\theta}_\star^s, \boldsymbol{x}_\star^s\}$, $\boldsymbol{d}_\star^s \in \chi^s$ in the source domain to be transferred to the target domain, we compute each Gaussian component's weights using Eq. (4),

$$\gamma_k = \frac{\pi_k \mathcal{N}(\boldsymbol{\theta}_\star^s \mid \boldsymbol{\mu}_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j \mathcal{N}(\boldsymbol{\theta}_\star^s \mid \boldsymbol{\mu}_j, \Sigma_j)}. \quad \ldots (11)$$

Finally, we map the query point $\boldsymbol{d}_\star^s$ as follows:

$$\hat{\boldsymbol{d}}_\star^t = \sum_{k=1}^{K} \gamma_k (B_k^{t\#} A_k \boldsymbol{z}_{k_\star}^s + \boldsymbol{\omega}_k^t), \quad \ldots (12)$$

where $\boldsymbol{z}_{k_\star}^s = B_k^s(\boldsymbol{d}_\star^s - \boldsymbol{\omega}_k^s)$ from Eq. (6).

For a given source trajectory, say a human demonstration $\xi_{1:T}^H = \{\boldsymbol{\theta}^H, \boldsymbol{x}^H\}_{1:T}$, transfer is performed for each point individually along the trajectory, using Eqs. (11) and (12), to obtain estimated target robot trajectories $\hat{\xi}_{1:T}^R = \{\hat{\boldsymbol{\theta}}^R, \hat{\boldsymbol{x}}^R\}_{1:T}$. Then each transferred trajectory can be encoded onto a parametrized skill as discussed below in Section 4.4.

A straightforward extension of our method to multi-robot systems involves applying the method separately for each robot to learn from the same human demonstrations. This approach, as illustrated in **Fig. 3**, assumes that all robots are learning at roughly the same time, in parallel from the same human demonstrations. This requires learning an LPA model for each teacher-learner pair.

Teaching multi-robot systems is attractive for complex tasks that require collaboration of multiple robots. Such tasks can be solved more easily by combining the unique capabilities of each robot, or faster by extending the area of coverage and range of operation [46, 47].

There may be a case where only one robot is available to learn from a human demonstrator, and after some period of time a new robot becomes available to learn the same skills. This may be the case where a new robot is delivered to a factory with existing robots that have previously learned these skills or a *skilled* robot is shipped to a different location with other robots that must learn the same skills.

In these settings, if the existing robot has *mastered* the skills and a human teacher is not available to teach the new robot, it may be beneficial to transfer knowledge from the existing robot to the new one. In this approach, human demonstrations are transferred to the new robot via the existing robot. The existing robot can master the skills by (a) correction with human feedback [48–50], (b) refining and fine-tuning with reinforcement learning approaches [51, 52], or (c) its own experience, using techniques based on socially guided exploration for robot learning of motor skills [53].

To apply our method in this setting, we must learn a mapping from a human teacher to the robot that will then act as a teacher to other robot learners. Then we learn a mapping between the robot teacher and the robot learners using the same procedure described in our method (**Algorithm 4**), without the step of projecting the trajectories onto a skeletal model. The challenge is how to get a robot to mimic the movement of another robot, in the same way a human would as described in Section 4.2.

However, for our purpose of analyzing the error accumulation of transfer from a human to multiple robots in series, we assume that the robot teacher has successfully learned its kinematic models, or that the models are well understood analytically. This assumption is reasonable, since a robot deployed in some environment for an extended period of time would generate some data from which to learn its kinematics models. Examples of this setting can be found in developmental robotics and lifelong learning.

Under this assumption we can replace the human teacher with a robotic teacher and apply the same method to transfer knowledge between robots. We explore and analyze knowledge transfer between robots in this context experimentally in Section 5.2.

4.4. Skill Encoding

We use Dynamic Movement Primitives to encode joint space trajectories as parametrized policies $\pi_{\boldsymbol{\beta}}$ and demonstrate that we can recover useful skills from human demonstrations adapted using our method. A DMP is specified by a set of nonlinear differential equations with well-defined attractor dynamics [10]. For a single DoF trajectory θ, the DMP is defined as follows:

$$\tau \dot{z} = \alpha_z(\beta_z(\theta_g - \theta) - z) + g(x), \quad \ldots\ldots (13)$$

$$\tau \dot{\theta} = z, \quad \ldots\ldots (14)$$

$$\tau \dot{x} = -\alpha_x x, \quad \ldots\ldots (15)$$

where x is the phase variable, z is the auxiliary variable and θ_g is the desired goal of the movement. Parameters α_z, β_z, α_x and τ define the behaviour of this second order system. If the parameters are selected as $\tau > 0$, $\alpha_z = 4\beta_z > 0$ and $\alpha_x > 0$, then the dynamic system has a unique point attractor at $\theta = \theta_g$, $z = 0$. Given the initial condition $x(0) = 1$, Eq. (15) is solved analytically by $x(t) = \exp(-\alpha_x t/\tau)$. However, to implement different modulations of the DMP such as phase stopping [11], it is better to keep Eq. (15) as a differential equation.

The forcing term $g(x)$ is defined as a linear combination of radial basis functions, which enable the robot to follow any smooth point-to-point trajectory from the beginning of the movement θ_0 to the end configuration θ_g:

$$g(x) = \frac{\sum_{i=1}^{N} w_i \Psi_i(x)}{\sum_{i=1}^{N} \Psi_i(x)} x, \quad \ldots\ldots (16)$$

$$\Psi_i(x) = \exp(-h_i(x - c_i)^2). \quad \ldots\ldots (17)$$

Here c_i are the centers of the radial basis functions distributed along the trajectory and $h_i > 0$. For robots with more than one DoF, each degree is represented by Eqs. (13)–(14) with different w_i and θ_g, but with a common phase variable x and time constant τ as specified in Eq. (15). To approximate any smooth trajectory with a DMP, we need to estimate the weights w_i, time constant τ, and the goal configuration θ_g. τ is usually set to the duration of the movement, θ_g to the final configuration on the trajectory, while w_i are estimated from the training data (sampled positions, velocities and accelerations) using regression techniques. See [10] for more details.

The training data here is the estimated target robot trajectory $\{\hat{\boldsymbol{\theta}}^R\}_{1:T}$ in joint space, which is transferred from human demonstrations by LPA, and the policy parameters to be learned are the DMP weights $\boldsymbol{\beta} = \{w_i\}$. We compute the first and second derivatives of the adapted trajectory to obtain its velocities and accelerations.

4.5. Summary

This section summarizes the steps involved in learning from demonstrations using our method. We described in Section 4.2 how a sample of correspondences between the human and robot domains can be collected. This involved identifying a pair of joints between the domains, that are in correspondence for a particular class of tasks, and identifying how we can convert between them; and also locating the demonstrated task in the workspace of the robot learner. Then a sample of correspondences is collected by generating random robot data in the vicinity of the tasks to be performed and using an IK solver of the human model on each robot data point to generate a corresponding human data set.

Once the sample of correspondences is available, an LPA model is learned on the data, as described in Section 4.3. Then any sequence of points in the human domain involving the joints used when collecting the training data, can be transferred to the robot domain using the learned LPA model. Finally, the transferred trajectories can be encoded as parametrized skills as described in Section 4.4. We also discussed how the same procedure can be applied to transfer knowledge between robots by replacing a human teacher with a robot, where the robot teacher is assumed to have learned its kinematics models from experience.

5. Experiments

To evaluate our transfer method, we designed experiments in simulation to demonstrate and transfer trajectories from a 7-DoF arm of a human model, to two humanoid robots, each with 7-DoF arms, namely the Willow Garage PR2[6] and Meka M1,[7] shown in **Fig. 13**. **Table 1** shows the parameters (lengths) of their arms as well as those of the human model. Note that, in addition to the arm length difference, the robots also have different offsets between their links.

We demonstrated goal-directed tasks of writing letters, using the common 2D handwriting movement data set.[8] The letters are distributed, scaled and rotated such that they span the 3D task space, as shown in **Fig. 13**. We also analyzed the transfer of the demonstrated tasks from one robot to another robot, to evaluate the possibility of knowledge transfer between robots.

The data set contains handwritten letters from *A* to *Z*, with several demonstrations per letter. In our experiments we only used 5 demonstrations for each letter in **Fig. 13** as test data. To model human demonstrations, we used an IK solver on the left arm of the human kinematic model to trace the letters and obtain the tasks in joint space; however, these demonstrations could have been recorded by any motion capture system and adapted onto the human

6. http://www.willowgarage.com/pages/pr2/overview [Accessed June 26, 2017]
7. https://github.com/ahoarau/mekabot [Accessed June 26, 2017]
8. https://gitlab.idiap.ch/rli/pbdlib-matlab/tree/master/data/2Dletters [Accessed June 26, 2017]

(a) PR2. (b) Meka.

Fig. 13. Robots used in our experiments.

Table 1. Parameters (lengths) of human and robot models in mm. ⋆Estimated from URDF model.

Link	Human	PR2	Meka⋆
Upper arm	250	400	279
Forearm	250	321	322

model, using for example, the Master Motor Map (MMM) framework [34]. We then used our method to adapt these trajectories onto the left arms of the robots and encoded them using DMPs.

To evaluate our method we executed reproductions of the transferred tasks on the robots, by computing forward kinematics on the simulated robots, to obtain the adapted trajectories in task space, and compared these against ground-truth data. As ground-truth, we used an IK solver on the robots to trace the letters to obtain their corresponding joint-space trajectories, encoded them using DMPs and executed their reproductions on the robots to obtain corresponding reproductions in task space. As a measure of performance, we computed the error in reproducing the letters in task space, using our method and the IK-based method.

When drawing small characters, the last three joints tend to move more whereas the first four stay relatively static. For the large characters in our experiments, the task lies almost entirely in the space of the first four joints, and can be reproduced in this 4-dimensional joint space. In this case, the domains χ^s and χ^t, data sets D^s and D^t, and trajectories ξ^s and ξ^t are 7-dimensional – 4 in joint space and 3 in task space. As described in Section 4.2, we identified correspondences between the 4 DoFs of the human and the PR2 as

$$\begin{bmatrix} \theta_1^{pr2} \\ \theta_2^{pr2} \\ \theta_3^{pr2} \\ \theta_4^{pr2} \\ 1 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \frac{\pi}{2} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_1^{human} \\ \theta_2^{human} \\ \theta_3^{human} \\ \theta_4^{human} \\ 1 \end{bmatrix}$$

and the Meka as

$$\begin{bmatrix} \theta_1^{meka} \\ \theta_2^{meka} \\ \theta_3^{meka} \\ \theta_4^{meka} \\ 1 \end{bmatrix} = \begin{bmatrix} -1 & 0 & 0 & 0 & \frac{\pi}{2} \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & \frac{\pi}{2} \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \theta_1^{human} \\ \theta_2^{human} \\ \theta_3^{human} \\ \theta_4^{human} \\ 1 \end{bmatrix}$$

where the 5×5 matrices are the posture mappings f_{post}^{pr2} and f_{post}^{meka} respectively.

To generate training data for learning the mapping for each robot, we simulated motor babbling by randomly sampling the 4-dimensional joint space of the robot and applying forward kinematics to obtain corresponding end-effector positions. As described in Section 4.2, we transformed the end-effector position of each robot data into the human frame using $T_R^{H^{-1}}$ and used an IK solver to collect the corresponding human data point, where the IK was initialized using human poses similar to corresponding robot poses generated using f_{post}^{pr2} and f_{post}^{meka} as described above. The rest of the joints (last three joints) were kept at constant values. The workspace alignment matrices T_{pr2}^H and T_{meka}^H were computed using the rough alignment algorithm using f_{post}^{pr2} and f_{post}^{meka} to map data in the joint spaces as described in Section 4.2.

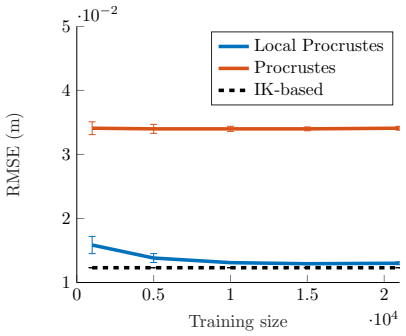
We start by evaluating the accuracy of mapping the trajectories from the human model onto the robots using Procrustes Analysis and Local Procrustes Analysis and encoding them using DMPs in Section 5.1, and finally we analyze the transfer of acquired knowledge by one robot to another robot in Section 5.2.

5.1. Skill Transfer and Encoding

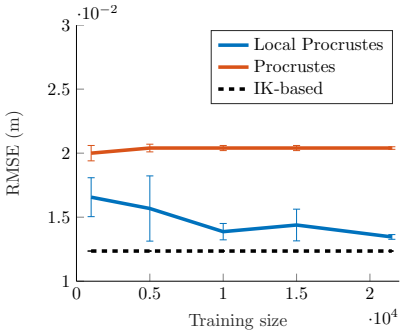
In this section we evaluate our method by analyzing the mapping of 5 human demonstrations per letter onto the robots, using Procrustes Analysis and Local Procrustes Analysis, and reproducing the corresponding encoded skills on the robots. We learned a DMP model for each letter from its 5 transferred demonstrations, and reproduced the first demonstration of each letter from these learned DMPs. To measure the transfer accuracy, the reproductions are compared against their corresponding original demonstrations. We repeated this for the IK-based method and averaged the results over 5 runs.

To train LPA (see **Algorithm 4**), we set $C_{\min}$ to 0.005, encouraging narrow clusters, and we experimented with several values of $N_{\min}$. Encouraging narrow clusters runs a risk of overfitting the mapping but this can be controlled by choosing a large value of $N_{\min}$. We found $N_{\min} = 15$ for the PR2 and $N_{\min} = 12$ for the Meka to learn more accurate mappings. Smaller values overfit the data and larger values produced less accurate mappings.

Figure 14 shows the accuracy in transferring and reproducing the letters, for PA and LPA, for increasing training sample size, compared to the IK-based method. The results of the IK method are shown in black, dashed lines.

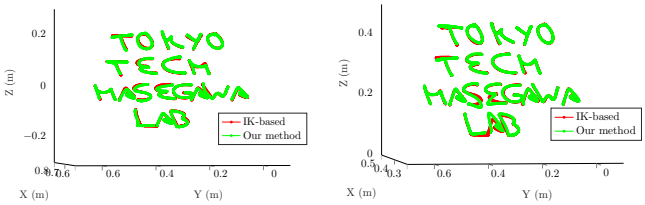


(a) Accuracy for the PR2.



(b) Accuracy for the Meka.

Fig. 14. Comparison of transferring using Procrustes Analysis and LPA, and the baseline method for the PR2 and Meka. The black dashed line corresponds to the results of the baseline method, and the error bars represent one standard deviation from the mean.



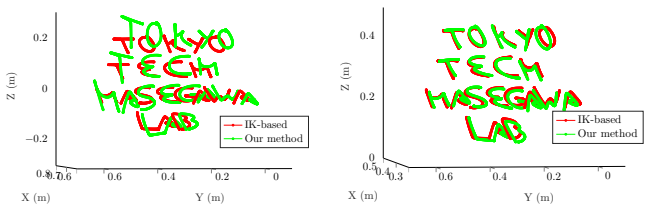
(a) Imitation for the PR2. (b) Imitation for the Meka.

Fig. 15. Sample task imitation for the PR2 and Meka using LPA.

We observe that transfer with LPA improves with increasing data points, and always out-performs PA. Transfer with PA does not improve with more data, as its linearity assumption limits it to model simple mappings.

Figure 15 shows some qualitative results of reproducing the transferred skills using LPA, for the PR2 (see **Fig. 15(a)**) and the Meka (see **Fig. 15(b)**), where the transferred trajectories are overlaid on the ground-truth trajectories. Similarly, **Fig. 16** shows the results of transfer using PA.

The reproductions using our method with LPA are comparable with the reproductions of the IK-based method, which makes use of kinematic models of the robots. Due to limited linear mappings, which resulted in less accurate transfer, the reproductions of our method using PA are offset from the IK-based reproductions. This is particularly clear for the PR2, where the PA mapping caused the final reproductions to differ significantly.



(a) Imitation for the PR2. (b) Imitation for the Meka.

Fig. 16. Sample task imitation for the PR2 and Meka using Procrustes Analysis.

Some letters with complex shapes, such as *H*, *K*, *A* and *B*, could not be reproduced exactly by DMP, accounting for much of the errors in **Fig. 14**, for both our method and the IK method. This is most likely because DMP aims to reproduce a generalized skill that can be adapted to new contexts, rather than an exact skill as demonstrated. Nevertheless, all the letters reproduced by the robots were recognizable, and the DMP was able to reproduce smooth generalized letters. The results presented here demonstrate that we are able to recover skills from adapted demonstrations, without assuming a kinematic model of the robot.

The results indicate that mapping with LPA adapts to the differences in kinematics, which requires less data for the PR2 than for the Meka. This difference is affected by the relative differences of the robots w.r.t. to the human model. It was observed in [29] that the complexity of the mapping required between manifolds of agents increases with the difference between the ratios of their arm links. The ratio of the first link to the second one of the human model is 1, and it is 1.25 for the PR2 and 0.87 for the Meka.

Although transfer with PA is unable to handle large differences in kinematics, it manages to preserve the overall gist of the demonstrated tasks. The trade off between LPA and PA is that LPA achieves better transfer accuracy given more data, but PA requires very few samples to learn the mapping. In the results presented here, the smallest sample size was 1000, however we observed that PA is able to learn a mapping with the same accuracy as reported here from a sample size as small as 500.

5.2. Knowledge Transfer Between Robots

In the previous experiment we demonstrated the transfer of skills from a human teacher to two robot learners. In this experiment we analyze the transfer of knowledge acquired from a human, between robots, where the PR2 is the teacher and the Meka is the learner, using LPA. In order to analyze this using our method, we assume that the teacher robot either has successfully learned its kinematics models, or that they can be modeled analytically. This is compared to the direct transfer from a human teacher to a new robot, the Meka, as presented in the previous experiment.

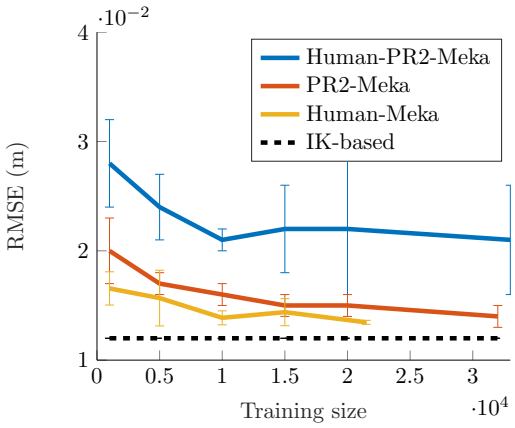
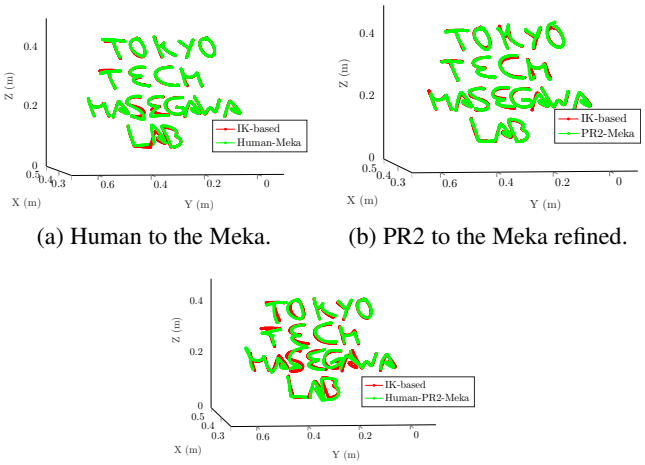


Fig. 17. Accuracy of transferring tasks of writing letters from the PR2 and human teacher to the Meka.



(a) Human to the Meka. (b) PR2 to the Meka refined.

(c) PR2 to the Meka unrefined.

Fig. 18. Transferred tasks reproduced by the Meka.

Firstly we took trajectories mapped to the PR2 from human demonstrations and mapped them to the Meka, thus effectively cascading the human-PR2 mapping and the PR2-Meka mapping. This is denoted *Human-PR2-Meka*. Separately, we assume the PR2 has mastered the skills transferred from a human, and transferred these *mastered* skills to the Meka, denoted *PR2-Meka*. We discussed some techniques which a robot can use to improve or refine skills learned from human demonstrations in Section 4.3.3. In this paper, we assume any of these techniques can be used, and thus model *mastered* robot skills using an IK solver on the robot teacher, the same way we modeled human demonstrations. These two approaches are compared with direct transfer from the human teacher to the Meka, denoted *Human-Meka*.

Figure 17 shows the comparison of transferring to the Meka using the three approaches. Direct transfer from human demonstrations is taken from the previous section. We observe that transferring refined skills via the PR2 is not as effective as direct transfer for the same amount of training data. This is due to the kinematics differences between the two robots being larger, compared to the difference between the Meka and the human model, indicated by the ratio of their links discussed in the previous section. However, given more training data, transferring refined skills via the PR2 becomes as effective as direct human transfer. Transferring unrefined skills via the PR2 is even less accurate because of the accumulation of errors from the human domain.

In **Fig. 18**, we show some qualitative results of transferring to the Meka, using the three approaches. The generalized letters reproduced are recognizable and comparable to the IK method. **Table 2** summarizes the results, where we show the minimum errors achieved by each approach. Direct transfer from the human teacher and transfer of refined skills from the PR2 are within the standard deviations of each other, and therefore are not statistically significantly different from each other. On the other hand, transferring unrefined skills from the PR2 is the least accurate and further refinement of the skills by the robot learner would be needed.

Table 2. Comparison of errors for transferring to the Meka, from human and the PR2. The numbers in brackets are standard deviations.

Source	Human	Human-PR2 refined	Human-PR2 unrefined
mapping (m)	0.0045 (0.0008)	0.0062 (0.0041)	0.0152 (0.0035)
encoding (m)	0.0140 (0.0013)	0.0138 (0.0005)	0.0192 (0.0012)

5.3. Discussion

The results presented in Section 5.1 demonstrated the efficacy of our proposed data-driven approach in adapting human demonstrations onto the arms of two humanoid robots with different kinematic parameters; and confirmed that skills on the robots can be successfully recovered from the adapted demonstrations. In contrast to the standard approach taken by other methods, we showed that generalized forms of the tasks can be reproduced by the robots, while preserving the goal-directed characteristics of the tasks, i.e., preserving the shape and size of the letters, rather than mimicking the posture of the teacher.

An interesting finding that requires further investigation is that, although complex non-linear mappings are generally preferred in most transfer learning problems for robotics, such as shared Autoencoders, Shared-GPLVM [16] and LPA [22, 29], simple linear mappings such as Procrustes Analysis could be more beneficial in a case where human demonstrations are used to provide initialization for learning with reinforcement learning for humanoids [51, 52], or a case where a human is allowed to provide feedback to the robot learner for correcting its reproductions [48–50]. This is because they can learn mappings from very few samples, which is desired for physical robots, and that they preserve the overall gist of the transferred skills, which would guide a reinforcement learner towards relevant spaces for exploration.

Results presented in Section 5.2 demonstrated the pos-

sibility of human-robot knowledge transfer in a multi-robot setting, where one robot acquired knowledge from a human teacher and then acted as a teacher to a new robot. Due to the accumulation of errors when transferring from a human to the new robot via the existing one, our results show that it is better for the robot acting as a teacher to first improve its skills, or alternatively the new robot can further refine its skills after transfer, where the transferred knowledge acts as prior knowledge for accelerating the learning process of the new robot.

6. Conclusion

This paper proposed a data-driven LfD approach for humans to easily transfer goal-directed skills to humanoid robots without knowledge of their kinematics. Experiments conducted on two simulated humanoid robots demonstrated that this approach is able to adapt human demonstrations onto the robots, and that useful skills can be recovered from the adapted demonstrations using DMPs. Furthermore, experimental results also showed that the skills learned by one robot from a human teacher can also possibly be transferred to another robot, in cases where a human teacher is not available.

A possible extension of our work is applying it to transfer human skills to non-anthropomorphic robots, where the challenge is that correspondences with the human body are not obvious. This would require that we revise our workspace alignment approach, which relies on the availability of correspondences between the human and the robot learner (which is easy to determine for humanoids). Furthermore, relaxing the requirement of correspondences, and considering non-linear mappings that do not rely on pairwise correspondences, would aid our approach to transfer knowledge to robots with arbitrary configurations.

The tasks used in our experiments allowed us to model non-linear mappings in a lower-dimensional subspace which was easy to determine. However, in other tasks in general it may not be obvious, thus requiring non-linear techniques that project the tasks onto some latent space. Investigating techniques such as those based on shared latent spaces (e.g., Shared-GPLVM) for learning mappings from random data is an interesting direction, or a combination of a non-linear dimensionality reduction technique with LPA. However, choosing the optimal dimensionality of the latent space is still a challenging problem. Lastly, for future work we aim to incorporate our method with reinforcement learning approaches for skill refinement for tasks that could not be accurately transferred, or alternatively to incorporate human feedback post-transfer.

Acknowledgements

This research was supported by Core Research for Evolutional Science and Technology (CREST) program of Japan Science and Technology Agency (JST). The authors would like to thank the reviewers for their helpful and insightful comments.

References:

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," Robotics and Autonomous Systems, Vol.57, Issue 5, pp. 469-483, 2009.

[2] R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction," Vol.1, Cambridge: MIT Press, 1998.

[3] M. Rolf, J. J. Steil, and M. Gienger, "Goal babbling permits direct learning of inverse kinematics," IEEE Trans. on Autonomous Mental Development, Vol.2, No.3, pp. 216-229, 2010.

[4] J. H. Figueroa Heredia, H. Sahloul, and J. Ota, "Teaching mobile robots using custom-made tools by a semi-direct method," J. of Robotics and Mechatronics, Vol.28, No.2, pp. 242-254, 2016.

[5] S. Calinon, "A tutorial on task-parameterized movement learning and retrieval," Intelligent Service Robotics, Vol.9, Issue 1, pp. 1-29, 2016.

[6] G. Maeda, M. Ewerton, D. Koert, and J. Peters, "Acquiring and Generalizing the Embodiment Mapping from Human Observations to Robot Skills," IEEE Robotics and Automation Letters, Vol.1, Issue 2, pp. 784-791, 2016.

[7] C. Nehaniv and K. Dautenhahn, "Like me? – measures of correspondence and imitation," Cybernetics and Systems, Vol.32, Issue 1-2, pp. 11-51, 2001.

[8] S. Calinon, F. D'halluin, E. Sauser, D. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation: An approach based on Hidden Markov Model and Gaussian Mixture Regression," IEEE Robotics and Automation Magazine, Vol.17, No.2, pp. 44-54, 2010.

[9] S. Khansari-Zadeh and A. Billard, "Learning stable nonlinear dynamical systems with Gaussian mixture models," IEEE Trans. on Robotics, Vol.27, No.5, pp. 943-957, 2011.

[10] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," Neural computation, Vol.25, No.2, pp. 328-373, 2013.

[11] A. Ude, B. Nemec, T. Petrić, and J. Morimoto, "Orientation in Cartesian space dynamic movement primitives," IEEE Int. Conf. on Robotics and Automation (ICRA), Hong Kong, pp. 2997-3004, 2014.

[12] N. H. Pham and T. Yoshimi, "Adaptive learning of hand movement in human demonstration for robot action," J. of Robotics and Mechatronics, Vol.29, No.5, pp. 919-927, 2017.

[13] B. Dariush, M. Gienger, A. Arumbakkam, C. Goerick, Youding Zhu, and K. Fujimura, "Online and markerless motion retargeting with kinematic constraints," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 191-198, 2008.

[14] T. Tosun, R. Mead, and R. Stengel, "A general method for kinematic retargetting: Adapting poses between humans and robots," Proc. Int. Mechanical Engineering Congress and Exposition (IMECE), pp. 1-10, 2014.

[15] C. Stanton, A. Bogdanovych, and E. Ratanasena, "Teleoperation of a humanoid robot using full-body motion capture, example movements, and machine learning," Proc. of Australasian Conf. on Robotics and Automation (ACRA), 2012.

[16] B. Delhaisse, D. Esteban, L. Rozo, and D. Caldwell, "Transfer Learning of Shared Latent Spaces between Robots with Similar Kinematic Structure," Proc. of the Int. Joint Conf. on Neural Networks, pp. 4142-4149, 2017.

[17] A. Shon, K. Grochow, and R. Rao, "Robotic imitation from human motion capture using gaussian processes," IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), pp. 129-134, 2005.

[18] N. Makondo, J. Claassens, N. Tlale, and M. Braae, "Geometric technique for the kinematic modeling of a 5 DOF redundant manipulator," 5th IEEE Conf. on Robotics and Mechatronics (ROBMECH), pp. 1-7, 2012.

[19] J. Sturm, C. Plagemann, and W. Burgard, "Body schema learning for robotic manipulators from visual self-perception," J. of Physiology – Paris, Vol.103, Issue 3-5, pp. 220-231, 2009.

[20] N. G. Tsagarakis, S. Morfey, G. M. Cerda, L. Zhibin, and D. G. Caldwell, "COMpliant huMANoid COMAN: Optimal joint stiffness tuning for modal frequency control," IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 673-678, 2013.

[21] G. Metta, G. Sandini, D. Vernon, L. Natale, and F. Nori, "The iCub humanoid robot: an open platform for research in embodied cognition," Proc. of the Workshop on Performance Metrics for Intelligent Systems (PerMIS), pp. 50-56, 2008.

[22] M. Hiratsuka, N. Makondo, B. Rosman, and O. Hasegawa, "Trajectory Learning from Human Demonstrations via Manifold Mapping," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), pp. 3935-3940, 2016.

[23] S. Tak and H. S. Ko, "A Physically-Based Motion Retargeting Filter," ACM Trans. on Graphics, Vol.24, Issue 1, pp. 98-117, 2005.

[24] M. Do, P. Azad, T. Asfour, and R. Dillmann, "Imitation of human motion on a humanoid robot using non-linear optimization," IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), pp. 545-552, 2008.
[25] L. Zhang, Z. Cheng, Y. Gan, G. Zhu, P. Shen, and J. Song, "Fast Human Whole Body Motion Imitation Algorithm for Humanoid Robots," IEEE Int. Conf. on Robotics and Biomimetics (ROBIO), pp. 1430-1435, 2016.
[26] A. Shon, K. Grochow, A. Hertzmann, and R. Rao, "Learning shared latent structure for image synthesis and robotic imitation," Adv. Neural Information Processing Systems (NIPS), pp. 1233-1240, 2006.
[27] T. Hirose and T. Taniguchi, "Abstraction Multimodal Low-Dimensional Representation from High-Dimensional Posture Information and Visual Images," J. of Robotics and Mechatronics, Vol.25, No.1, pp. 80-88, 2013.
[28] M. Field, D. Stirling, and Z. Pan, "Learning Trajectories for Robot Programing by Demonstration Using a Coordinated Mixture of Factor Analyzers," IEEE Trans. on Cybernetics, Vol.46, Issue 3, pp. 706-717, 2016.
[29] N. Makondo, B. Rosman, and O. Hasegawa, "Knowledge Transfer for Learning Robot Models via Local Procrustes Analysis," IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), pp. 1075-1082, 2015.
[30] B. Bócsi, L. Csató, and J. Peters, "Alignment-based Transfer Learning for Robot Models," Proc. of the IEEE Int. Joint Conf. on Neural Networks (IJCNN), pp. 1-7, 2013.
[31] S. J. Pan and Q. Yang, "A Survey on Transfer Learning," IEEE Trans. on Knowledge and Data Engineering, Vol.22, No.10, pp. 1345-1359, Oct. 2010.
[32] P. Azad, T. Asfour, and R. Dillmann, "Toward an Unified Representation for Imitation of Human Motion on Humanoids," IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 2558-2563, 2007.
[33] S. Gärtner, M. Do, and T. Asfour, "Generation of human-like motion for humanoid robots based on marker-based motion capture data," Proc. Int. Symp. Robotics (ROBOTIK), pp. 1-8, 2010.
[34] Ö. Terlemez, S. Ulbrich, C. Mandery, M. Do, N. Vahrenkamp, and T. Asfour, "Master Motor Map (MMM) – Framework and Toolkit for Capturing, Representing, and Reproducing Human Motion on Humanoid Robots," IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), pp. 894-901, 2014.
[35] C. Mandery, Ö. Terlemez, M. Do, N. Vahrenkamp, and T. Asfour, "Unifying Representations and Large-Scale Whole-Body Motion Databases for Studying Human Motion," IEEE Trans. on Robotics, Vol.32, Issue 4, pp. 796-809, 2016.
[36] J. Koenemann, F. Burget, and M. Bennewitz, "Real-time Imitation of Human Whole-Body Motions by Humanoids," IEEE Int. Conf. on Robotics and Automation (ICRA), pp. 2806-2812, 2014.
[37] C. Moulin-Frier and P. Y. Oudeyer, "Exploration strategies in developmental robotics: A unified probabilistic framework," IEEE 3rd Joint Int. Conf. on Development and Learning and Epigenetic Robotics (ICDL-Epirob), pp. 1-6, 2013.
[38] C. Wang and S. Mahadevan, "Manifold alignment using Procrustes analysis," Proc. of the 25th Int. Conf. on Machine learning (IJCAI), pp. 1120-1127, 2008.
[39] C. Wang and S. Mahadevan, "Manifold alignment without correspondence," Int. Joint Conf. on Artificial Intelligence (IJCAI), pp. 1273-1278, 2009.
[40] F. Diaz and D. Metzler, "Pseudo-aligned multilingual corpora," The Int. Joint Conf. on Artificial Intelligence (IJCAI) pp. 2727-2732, 2007.
[41] D. Zhai, B. Li, H. Chang, S. Shan, X. Chen, and W. Gao, "Manifold alignment via corresponding projections," Proc. of the British Machine Vision Conf., pp. 3.1-3.11, 2010.
[42] Z. Cui, H. Chang, S. Shan, and X. Chen, "Generalized unsupervised manifold alignment," Advances in Neural Information Processing Systems, pp. 2429-2437, 2014.
[43] C. G. Atkeson, A. W. Moore, and S. Schaal, "Locally weighted learning for control," Artificial Intelligence Review, Vol.11, pp. 75-113, 1997.
[44] J. Ting, S. Vijayakumar, and S. Schaal, "Locally weighted regression for control," Encyclopedia of Machine Learning, pp. 613-624, Springer US, Boston MA, 2010.
[45] J. Verbeek, "Learning nonlinear image manifolds by global alignment of local linear models," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.28, No.8, pp. 1236-1250, 2006.
[46] S. Chernova and M. Veloso, "Confidence-based multi-robot learning from demonstration," Int. J. of Social Robotics, Vol.2, No.2, pp. 195-215, 2010.
[47] J. H. Figueroa Heredia, J. I. U. Rubrico, S. Shirafuji, and J. Ota, "Teaching tasks to multiple small robots by classifying and splitting a human example," J. of Robotics and Mechatronics, Vol.29, No.2, pp. 419-433, 2017.
[48] S. Calinon and A. Billard, "Active teaching in robot programming by demonstration," RO-MAN 2007 – The 16th IEEE Int. Symp. on Robot and Human Interactive Communication, pp. 702-707, Jeju, 2007.
[49] S. Griffith, K. Subramanian, J. Scholz, C. Isbell, and A. L. Thomaz, "Policy Shaping: Integrating human feedback with reinforcement learning," Advances in Neural Information Processing Systems, pp. 2625-2633, 2013.
[50] R. Loftin, B. Peng, J. MacGlashan, M. L. Littman, M. E. Taylor, J. Huang, and D. L. Roberts, "Learning behaviors via human-delivered discrete feedback: modeling implicit feedback strategies to speed up learning," Autonomous Agents and Multi-Agent Systems, Vol.30, No.1, pp. 30-59, 2016.
[51] M. E. Taylor, H. B. Suay, and S. Chernova, "Integrating reinforcement learning with human demonstrations of varying ability," Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS), Tumer, Yolum, Sonenberg and Stone (Eds.), pp. 617-624, Taipei, Taiwan, May 26, 2011.
[52] T. Brys, A. Harutyunyan, V. U. Brussel, and M. E. Taylor, "Reinforcement learning from demonstration through shaping," Proc. of 24th Int. Joint Conf. on Artificial Intelligence (IJCAI), pp. 3352-3358, 2015.
[53] S. M. Nguyen and P. Y. Oudeyer, "Socially guided intrinsic motivation for robot learning of motor skills," Autonomous Robots, Vol.36, No.3, pp. 273-294, 2014.

Supporting Online Materials:

[a] https://github.com/ahoarau/mekabot/wiki/Meka-robot-overview [Accessed June 26, 2017]

Name:
Ndivhuwo Makondo

Affiliation:
Tokyo Institute of Technology
Council for Scientific and Industrial Research (CSIR South Africa)

Address:
4259 Nagatsuta-cho, Midori-ku, Yokohama 226-8503, Japan

Brief Biographical History:
2007-2013 B.Sc. and M.Sc. Electrical Engineering, University of Cape Town
2011-2013 Masters Studentship, Council for Scientific and Industrial Research
2013- Research and Development Engineer, Council for Scientific and Industrial Research
2013- Doctoral Student, Tokyo Institute of Technology

Main Works:
• M. Hiratsuka, N. Makondo, B. Rosman, and O. Hasegawa, "Trajectory Learning from Human Demonstrations via Manifold Mapping," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2016.
• N. Makondo, B. Rosman, and O. Hasegawa, "Knowledge Transfer for Learning Robot Models via Local Procrustes Analysis," IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids), 2015.

Name:
Michihisa Hiratsuka

Affiliation:
Data Engineering Group, Recruit Lifestyle Co., Ltd.

Address:
1-9-2 Marunouchi, Chiyoda-ku, Tokyo 100-6640, Japan

Brief Biographical History:
2013 B.Sc. Eng. in System Design Engineering, Keio University
2016 M.Sc. Eng. in Computational Intelligence and Systems Science, Tokyo Institute of Technology
2016- Data Engineering Group, Recruit Lifestyle Co., Ltd.

Main Works:
• M. Hiratsuka, N. Makondo, B. Rosman, and O. Hasegawa, "Trajectory Learning from Human Demonstrations via Manifold Mapping," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS), 2016.

Name:
Benjamin Rosman

Affiliation:
Principal Researcher, Mobile Intelligent Autonomous Systems, CSIR

Address:
A18, Building 17a, CSIR, Meiring Naude Road, Brummeria, Pretoria 0001, South Africa

Brief Biographical History:
2009-2014 Ph.D., University of Edinburgh
2014- Researcher, Council for Scientific and Industrial Research (CSIR)
2014- Lecturer, University of the Witwatersrand

Main Works:
• B. Rosman, M. Hawasly, and S. Ramamoorthy, "Bayesian Policy Reuse," Machine Learning J., Vol.104, No.1, 2016.
• B. Rosman and S. Ramamoorthy, "Action Priors for Learning Domain Invariances," IEEE Trans. on Autonomous Mental Development, 2015.
• B. Rosman and S. Ramamoorthy, "Learning Spatial Relationships between Objects," The Int. J. of Robotics Research, Vol.30, No.11, 2011.
• A. Saxe, A. Earle, and B. Rosman, "Hierarchy Through Composition with Multitask LMDPs," Int. Conf. on Machine Learning, 2017.
• N. Makondo, B. Rosman, and O. Hasegawa, "Knowledge Transfer for Learning Robot Models via Local Procrustes Analysis," IEEE-RAS Int. Conf. on Humanoid Robots, 2015.
• P. Ranchod, B. Rosman, and G. Konidaris, "Nonparametric Bayesian Reward Segmentation for Skill Discovery Using Inverse Reinforcement Learning," IEEE/RSJ Int. Conf. on Intelligent Robots and Systems, 2015.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE)

Name:
Osamu Hasegawa

Affiliation:
Associate Professor, Tokyo Institute of Technology
CEO, SOINN Inc.

Address:
Rm 1418, J3-13, 4259 Nagatsuta-cho, Midori-ku, Yokohama 226-8503, Japan

Brief Biographical History:
1993 Received Dr. Eng. degree in Electronic Engineering from The University of Tokyo
1993-1999 Research Scientist, Electrotechnical Laboratory
2000-2002 Visiting Scientist, Robotics Institute, Carnegie Mellon University
2002- Faculty Member, Imaging Science and Engineering Laboratory, Tokyo Institute of Technology
2014- CEO, SOINN Inc.

Main Works:
• S. Furao and O. Hasegawa, "An incremental network for on-line unsupervised classification and topology learning," Neural Networks, Vol.19, No.1, pp. 90-106, 2006.
• T. Toyoda and O. Hasegawa, "Random field model for integration of local information and global information," IEEE Trans. on Pattern Analysis and Machine Intelligence, Vol.30, No.8, pp. 1483-1489, 2008.
• N. Makondo, B. Rosman, and O. Hasegawa, "Knowledge Transfer for Learning Robot Models via Local Procrustes Analysis," IEEE-RAS Int. Conf. on Humanoid Robots, 2015.
• A. Sudo, A. Sato, and O, Hasegawa, "Associative memory for online learning in noisy environments using self-organizing incremental neural network," IEEE Trans. on Neural Networks, Vol.20, No.6, pp. 964-972. 2009.

Membership in Academic Societies:
• The Institute of Electrical and Electronics Engineers (IEEE) Computer Society
• The Institute of Electronics, Information and Communication Engineers (IEICE)
• Information Processing Society of Japan (IPSJ)