A Morphable Template Framework for Robot Learning by Demonstration: Integrating One-shot and Incremental Learning Approaches

Yan Wu[a,c], Yanyu Su[b,c], and Yiannis Demiris[c]

[a] *Robotics Programme, A*STAR Institute for Infocomm Research, Singapore 138632*
[b] *State Key Laboratory of Robotics and System, Harbin Institute of Technology, China*
[c] *Electrical & Electronic Engineering Dept, Imperial College, London SW7 2AZ, UK*

Abstract

Robot learning by demonstration is key to bringing robots into daily social environments to interact with and learn from human and other agents. However, teaching a robot to acquire new knowledge is a tedious and repetitive process and often restrictive to a specific setup of the environment. We propose a template-based learning framework for robot learning by demonstration to address both generalisation and adaptability. This novel framework is based upon a one-shot learning model integrated with spectral clustering and an online learning model to learn and adapt actions in similar scenarios. A set of statistical experiments is used to benchmark the framework components and shows that this approach requires no extensive training for generalisation and can adapt to environmental changes flexibly. Two real-world applications of an iCub humanoid robot playing the tic-tac-toe game and soldering a circuit board are used to demonstrate the relative merits of the framework.

Keywords: imitation, learning by demonstration, template warping, learning primitive actions, robot soldering

1. Introduction

In the last few decades, robot learning by demonstration (LbD) has become one of the most active research topics in robotics. It encapsulates the imitation capability of a robot to perceive, learn, reproduce and adapt according to environmental changes the motor skills for a given task demonstrated by another agent with extrinsic parameters [1, 2], such as learning to solder (Figure 1).

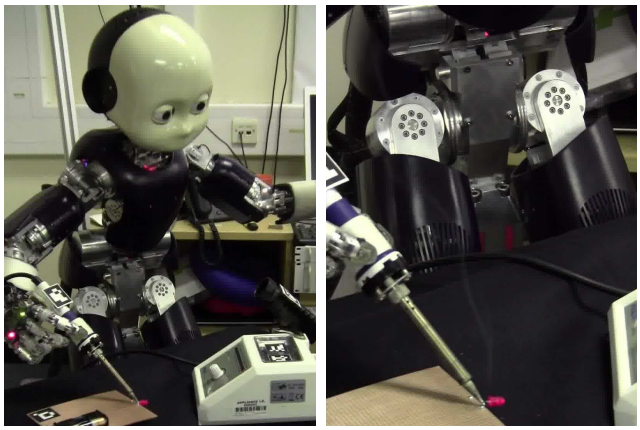


Figure 1: The iCub humanoid robot soldering a circuit component.

Owing to its relative merits over traditional methodologies for robot programming primarily by offering a user-friendly teaching framework to programme robots, LbD has spawned an array of research directions to make robots interact in social environments [3, 4, 5, 6, 7, 8, 9, 2]. Many of these works attempt to construct learning models to answer some of the 5 LbD "W"s [10]: In a natural learning environment, it can be a cluttered scene with multiple people and objects with movements happening at different

times and places. Thus, identifying **w**ho to imitate at **w**here and **w**hen on **w**hat body-parts and objects as well as learning the actual ho**w** to imitate are some of the key research questions.

In this paper, we present a morphable template learning framework to address two "how-to" questions in LbD - namely how to reduce learning fatigue and how to adapt the learning result to a similar but new environment. This work incorporates online learning into a morphable one-shot learning model to construct an incremental learning framework for action imitation, such as the soldering task presented below. It segments a demonstration of action into a sequence of basic trajectories and stores them as templates. Each template is generalised and updated using an online learning model when multiple demonstrations of an action are performed. To generate a learned action in a new environment, plan adaptation is used to map the template to the new situation. As the "what-to" question is not the focus of this work, we avoid the correspondence problem [11] by using colour blobs, SIFT features[12] and AR markers in our experiments.

The remainder of this paper is organised as follows: In Section 2, we present some related work in robot learning by demonstration. Section 3 introduces algorithms that are adopted in this work. In Section 4, the overview of our proposed framework is introduced followed by the detailed description of its building blocks. In Section 5, we describe the experiments to statistically evaluate our framework using a dataset of goal-oriented hand trajectories. Finally, two real-world applications of the iCub humanoid robot using this framework are presented in Section 6 before Section 7 concludes this paper with the overview of future work.

2. Related Work

Most LbD work can be categorised into either learning a mapping function to approximate the state-action relationship or learning a system model to represent the world dynamics [13]. The system model approach typically involves reinforcement learning to find a policy from demonstrations for relating its action and the world dynamics. While this approach is promising, the learning complexity scales exponentially with respect to the number of observed features which greatly increases learning fatigue and outcome uncertainty.

The mapping function approach, which uses demonstrations to establish a direct mapping from the state observations to the required actions, usually performs batch learning, incurring no additional cost for model update in between executions. The state-of-the-art mapping function approaches in LbD investigate the use of probabilistic generative models, such as Gaussian Mixture Regression (GMR) [14, 15], Hidden Markov Models (HMM) [16] and Gaussian Process Regression (GPR) [2]. In particular, variants of GMR have been shown to be useful in many LbD applications, such as HMM-GMR [1] and Dirichlet Process GMR (DP-GMR) [13]. However, to update these supervised learning models, re-training of the model is required. This computationally expensive property hinders long-term interactions in a daily set up, and the learned action generalised from demonstrations is highly restricted to a specific setting. Thus, simple copying and normal function approximation approaches may not always work well in reproducing observed trajectories in an unseen situation. Moreover, additional constraints may be present in a new situation, such as forced waypoints that require some

minor modification to the observed action. Hence, a robot must possess the ability to adapt to environmental variations for the same actions while maintaining its generalisation capability. Previous work in the literature focuses on approaches either to produce a unique but exactly-corresponding imitation of the previously demonstrated action [17, 18] or to generate a new trajectory based on a subset of the competent tasks to accommodate additional constraints [19, 20, 21].

In order to work successfully, most LbD frameworks require first the task to be demonstrated several times [19, 22]. This time-consuming and tedious process is not favoured by human demonstrators. To reduce fatigue and accelerate the learning process in giving demonstrations will be crucial for future robotic advances. Thus, one-shot learning [23], a popular niche area in machine intelligence becomes a favourable approach in tackling this problem [24, 25]. Some of this work focuses on explanation-based methods [26] which use prior knowledge to explain how each observation satisfies the target concept. This highly symbolic-reasoning approach is difficult to be implemented on physical robots without hard-coding a large repertoire of action primitives. Furthermore, one-shot learning methods alone do not address the issue of generalisation should multiple demonstrations of an action subsequently appear before the robot. A more sophisticated model based on one-shot learning approach can be an answer to handle generalisation.

Friedrich et al. [27] argue that approaches generalising observations into a set of intrinsic complex model statistics or internal states limit the ability of user interaction with the model after demonstration. However, many state-of-the-art paradigms make use of such models. For example, in the

application of Gaussian mixture models [28, 29], the generalised parameters are the weights with associated Gaussian parameters. These approaches deter direct user intervention/interaction, as the users cannot easily interpret and manipulate the implicit meaning of the parameters. When a robot is deployed in a social environment for continuous learning, this becomes of cardinal importance because users are not able to help robots to learn from mistakes.

Many established LbD models focus on learning a task as one single action [9, 30]. Actions learned with these models can be performed accurately but more difficult to be partially reused in a novel scenario. Thus, [31] and [32] argue for the representation of a task action by multiple movement primitives as a prerequisite for imitation learning based upon biological evidence. By segmenting an action into a sequence of basic movement primitives [33], this representation expresses a task action as a chain-event of primitives similar to the hierarchical action control structure in the human motor cortex [34]. However, in various primitive-based LbD models [35, 36, 37], the list of primitives is exhaustive and hand-coded for a particular set of actions which has limited application in an unseen task. Another approach is to extract the motion constraints as a criterion for segmentation [38]. However, little literature to our knowledge addresses the issue of segmenting a primitive into a logical set of smaller actions at the learning phase and piecing them up at the action phase.

In image processing and motion planning, scenes are often warped from the current context into new ones using a morphing technique which minimises the bending energy [39, 40, 41]. This technique preserves the spatial

relationship between a set of features presents in both the original context and the new one. We believe that the adaptability issue in LbD could get inspiration from this approach. A trajectory imitation problem can be treated as a plan adaptation [25] which projects the demonstrated action with a set of environmental features into a similar but novel situation. This will generate a new trajectory by maintaining the spatial relations between the corresponding environmental features.

3. Background

In this section, we will briefly introduce three key models used in our learning framework, namely the thin-plate spline warping [42, 39] which is extended from its original concept to a 3D environment to address the adaptability issue, the online echo-state Gaussian process [43] which is used to generalised repeated demonstrations of an action and the spectral clustering algorithm [44] which is used for automatic action segmentation of a demonstration into smaller templates.

3.1. Thin-plate spline warping

Thin plate splines (TPS) were first introduced to geometric design and became popular in non-rigid transformation models in image processing and shape matching [42, 39]. As argued in Section 2, this template matching algorithm can be extended to an LbD problem. We refer to a given demonstration as a *template* and generation of the learned action in a new situation where the robot has to imitate as a *task*. An imitated trajectory can be seen as a warped demonstrated trajectory in this approach.

Given that there are k corresponding pairs of features present in both the template and the task (called invariant features, such as the locations of the target position), we can generate a distortional mapping to minimise the bending energy for each spatial location present in the template trajectory into a set of possible waypoints.

We define the k Cartesian invariant features in the template as the invariant control points (ICP) $\mathbf{P}$ and the corresponding ICPs in the task $\mathbf{P}'$, f as the mapping function from $\mathbf{P}$ to $\mathbf{P}'$. [42] shows that in a 2D scenario, to minimise the distortion of spatial features is equivalent to minimise the following energy function:

$$E = \sum_{w=1}^{k} \parallel \mathbf{P}'_w - \mathbf{P}_w \parallel + \lambda E_f \tag{1}$$

where

$$E_f = \int \int_{R^2} (f''_{xx} + 2f'_{xy} + f''_{yy}) dxdy \tag{2}$$

λ, the *regularisation parameter*, is introduced to balance the trade-off between the exact matching of points and the smoothness, which is particularly useful in the presence of noise as TPS is sensitive to noise. The mapping function f in a 2-D Cartesian space $f(x, y)$is defined as:

$$f(x, y) = \alpha_0 + \alpha_x x + \alpha_y y + \sum_{i=1}^{k} \omega_i \phi(\parallel (x_i, y_i) - (x, y) \parallel) \tag{3}$$

and

$$\phi(r) = r^2 log(r) \tag{4}$$

where (3) is a 2^{nd} order poly-harmonic spline also known as a Thin Plate Spline. To ensure the existence of E_f, the 2^{nd} derivatives of $f(x, y)$ must be

square integrable, which means that the following conditions have to be met:

$$\sum_{i=1}^{k} \omega_i = 0 \tag{5}$$

$$\sum_{i=1}^{k} \omega_i x_i = \sum_{i=1}^{k} \omega_i y_i = 0 \tag{6}$$

By letting $\Phi_{ij} = \phi(\| (x_i, y_i) - (x_j, y_j) \|)$ and $\upsilon_i = f(x_i, y_i)$, we can form a linear equation based on (4) - (6) as follows:

$$\begin{bmatrix} \mathbf{\Phi} & \mathbf{L} \\ \mathbf{L}^T & \mathbf{0} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega} \\ \boldsymbol{\alpha} \end{bmatrix} = \begin{bmatrix} \boldsymbol{\upsilon} \\ \mathbf{0} \end{bmatrix} \tag{7}$$

where $\boldsymbol{\omega}$ is a column vector of ω_i, $\boldsymbol{\alpha}$ is a column vector of $[\ \alpha_0 \quad \alpha_x \quad \alpha_y\ ]^T$ and the i^{th} row of $\mathbf{L}$, $\mathbf{L}_i = [\ 1 \quad x_i \quad y_i\ ]$.

Powell [45] show the non-singularity of the square matrix in (7). We can then define the upper left $k \times k$ sub-matrix of the inverse of this square matrix by $\mathbf{M'}_k$. It can be shown that $E_f \propto \boldsymbol{\upsilon}^T \mathbf{M'}_k \boldsymbol{\upsilon} = \boldsymbol{\omega}^T \mathbf{M} \boldsymbol{\omega}$. Thus, depending on the trade-off of accuracy and computational cost, the optimal solution of $\boldsymbol{\omega}$ and $\mathbf{a}$ with minimum bending energy can be solved either by analytical method or approximation methods described in [46] and [47].

3.2. Online echo state Gaussian process

While many LbD models use supervised learning techniques to achieve model generalisation, it presents a dilemma to be used in a real-world HRI environment where tedious training before the robot can learn a task may not be feasible. Moreover, once trained, these models are fixed until the next programmer-defined update. Such huge model updates are computationally

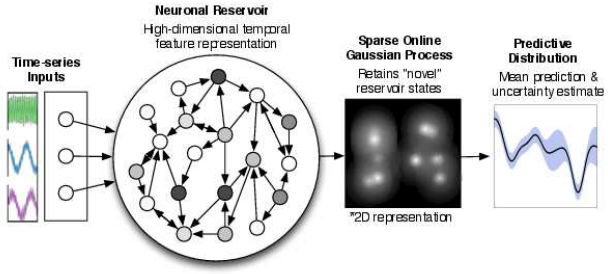


Figure 2: The Online Echo-State Gaussian Process [43]

expensive and the definition of when to update can be tricky. Thus, we require a model that can perform learning on the fly for spatial-temporal data. The online echo state Gaussian process (OESGP) proposed in [43] is a Bayesian-based online learning method that is capable of learning complex temporal relationships iteratively between sequential observations. It is a regular echo state network (ESN) trained with a sparse approximation of Gaussian process (GP). This allows the output prediction to contain both the mean and an uncertainty estimate. Moreover, kernels can be used to allow non-linear mappings between the reservoir states of the ESN and the outputs which helps modelling a wider range of dynamical systems.

Figure 2 is an illustration of the OESGP. When a given time-series data sequence is presented to the model, the state of the ESN is trained and updated. Then the updated ESN posterior is fed to the sparse online GP where the more informative states are retained in a closest GP and generates a mean prediction with an uncertainty estimate.

The non-parametric property of OESGP allows learning of temporal dynamics without having the problem of the GP growing unbounded while allowing the system to learn on the fly. Although the accuracy of the OESGP

depends on kernel selection, noise parameter setting and also the size of the "novelty" covariance matrix which affects the computational cost of the model, however, many successful benchmarks [43] in machine-learning and robotics together with the merits of the model suggests that it is a suitable candidate module in our LbD model.

3.3. Spectral clustering algorithm

When an action is demonstrated, unless the notion of primitive actions is given to the demonstrator, the demonstration usually comprises an uninterrupted series of subactions. This makes it difficult to "recycle" partial learned actions in a new scenario without segmenting the observation appropriately. If we assume that a demonstrated action consists of subactions which are separable in spatial-temporal manner, a suitable clustering algorithm can be used as an action segmentation algorithm. Like most of its counterparts, the original spectral clustering algorithm treats the temporal input just as an input feature. However, assuming time and space with the same "resolution" can lead to undesirable clustering results.

An adapted version of the spectral clustering algorithm was proposed in [44] to address this issue and achieve spatial scaling, rotation and action speed invariances. This property allows similar segmentation outcome for demonstrated actions performed by different users in different environment at different spatial scales. It can be used to generate primitive actions from demonstrations and eliminate the need to provide a list of exhaustive and hand-coded primitives.

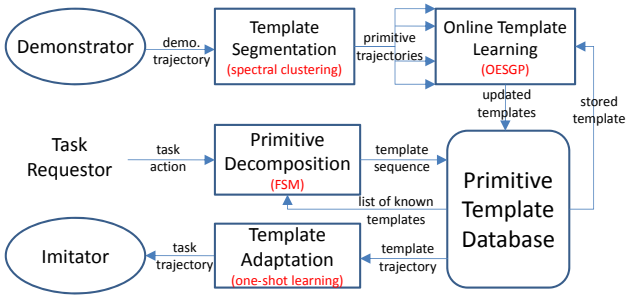


Figure 3: The schematics of the Morphable Template Learning Framework. This framework consists of four modules and a database which stores the labelled morphable templates. A demonstrated action is first segmented by Template Segmentation into a sequence of template trajectories. Any template trajectory demonstrated before will be updated by Online Template Learning and stored into the Primitive Template Database. When a task action is requested, it will be decomposed by Primitive Decomposition into a sequence of learned templates. With their associated invariant features, the Template Adaptation morphs the template trajectory into the required task trajectory.

4. The Morphable Template Learning Framework

This section presents our proposed LbD framework, a mapping function approach based on template adaptation integrated with an online learning paradigm [43], a spatial-temporal segmentation algorithm [44] and a Finite State Machine (FSM). In this work, we assume that all required input features are fully observable. This can be realised using either colour blob tracking, SIFT feature detection or AR markers to avoid the correspondence problem which is a research topic on its own. Our morphable template learning framework is built on the following four concepts and illustrated in Figure 3:

- The basic building block of Template Adaptation acts as an efficient module for generating an action given the *task* based on constraints

mapping with a *template*.

- A suitable template segmentation algorithm that segments a demonstrated *template* into a set of *sub-templates*.
- An online learning framework to update a stored *template*/*sub-template* with a new demonstration of the *template*.
- An algorithm that decomposes the *task* into an action plan made up from a sequence/hierarchy of *templates* and/or *sub-templates*.

4.1. Template Adaptation

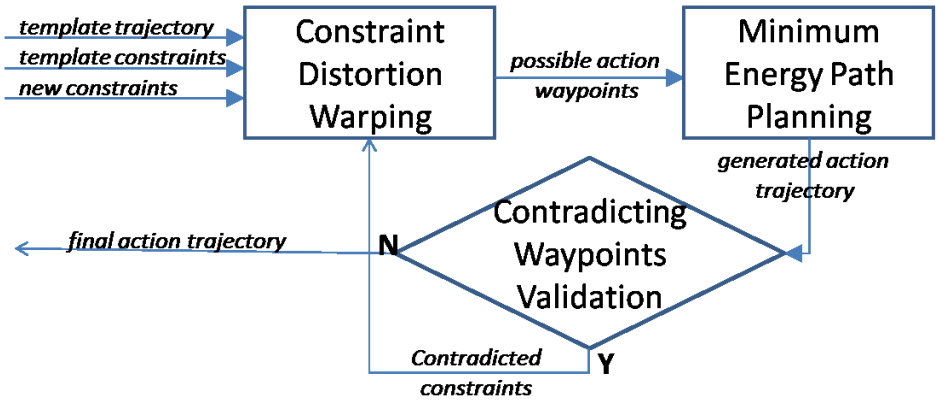


Figure 4: The schematics of Template Adaptation Module

We consider the case of trajectory planning in a 3D environment from a demonstration of a similar task. The ultimate aim of this module is to produce a desirable trajectory in a given scenario. This template adaptation algorithm consists of three components illustrated in Figure 4. In summary, it first generates a set of waypoints based on distortional warping of a set of corresponding constraints extracted from both a template and the task.

Using minimum-energy planning, a probable trajectory is then created from the time-series information associated with these possible locations. Finally, the generated path is subject to an *Interactive Plan Adjustment* for trajectory correction. This implementation is based on our previous work on one-shot imitation learning approach [25].

The use of a template based approach also has three main benefits: 1) it reduces demonstration fatigue by allowing LbD from the very first demonstration, as all useful demonstrations are stored as templates; 2) The templates and environmental features are human-readable; and 3) Templates can be easily broken down into sub-templates and used as primitive actions.

4.1.1. Definitions and Assumptions

For a given demonstration, we describe the motion trajectory as a set of p discrete spatial feature points $\mathbf{m}_l : (x_l, y_l, z_l), l \in \{1...p\}$ in time series obtained either from stereo-vision or motion capture devices. We also assume that the invariant feature points, including the target location in the scene can be fully described by a set, F of n point-like features, where each is described by $(\mathbf{a}_i, \mathbf{A}_i)$, $i \in \{1...n\}$. While $\mathbf{a}_i$ represents the Cartesian coordinates of the feature point, $\mathbf{A}_i$ captures the additional information that will help to match invariant points of similar objects/targets, such as SIFT [12] features and textual features.

In a novel situation described similarly by a set $F' : (\mathbf{a'}_j, \mathbf{A'}_j), j \in \{1...n'\}$, we assume that there exists an evaluation function of correspondence, $f_c(\mathbf{A}_i, \mathbf{A'}_j)$, where

$$f_c(\mathbf{A}_i, \mathbf{A'}_j) = \begin{cases} 0 & \text{if } \mathbf{A}_i \ \& \ \mathbf{A'}_j \text{ are uncorrelated} \\ 1 & \text{if } \mathbf{A}_i \text{ matches } \mathbf{A'}_j \end{cases} \tag{8}$$

Depending on the context, we can use different feature mapping algorithms to match $\mathbf{A}$s and $\mathbf{A'}$s, such as SIFT in image-based data. In this work, we use existing feature extraction tools for this correspondence problem. In the Section 5, we hard-code the features while SIFT matching and AR Markers are used respectively in Sections 6.1 and 6.2. We can then identify a maximum k pairs of Cartesian coordinates in the both the task space, $\mathbf{a'}_j$, and one of the learned templates, $\mathbf{a}_i$, where $f_c(\mathbf{A}_i, \mathbf{A'}_j) = 1$, $k \leq n$, $k \leq n'$. This k pairs of coordinates also include the pair of starting positions in the template and the task.

In a general path planning situation, we may be able to obtain more invariant features in both the task and the templates than the necessary set for mapping. Although more invariant features present in the scene help to guide the mapping more precisely, however, there are cases where inclusion of such features undesirably generates excessive output distortion. One example can be objects at a distance and textual features of the background. Detection of these invariant features is a problem in computer vision, and is beyond the scope of this work. In order for our work not to impose the matching constraints for excessive number of invariant features and hence to preserve the spatial relationship among the cardinal features, we filter out these features by setting a region of interest (ROI) for the observations. The generated trajectory for the task $\mathbf{m'} : (x, y, z)$ is then based on the mapping of the invariant features between the ROIs of the template and the task.

In a more complex situation, when the robot is required to route through

waypoints or to avoid obstacles, where an additional set $F'_x(\mathbf{b}_j, \mathbf{B}_j)$ can describe these extra feature points. If F'_x exists in a task, these additional feature points will only be used when the planned trajectory $\mathbf{m}'$ contradicts any $\mathbf{b}_j$. For example, if $\mathbf{b}_j$ represents a forced waypoint where the trajectory has to route through, when the generated trajectory does not pass through $\mathbf{b}_j$, this is considered as a contradiction. In the following sections, the mapping algorithm will be described in detail.

4.1.2. Constraint Distortion Warping

Section 3.1 describes the TPS in its 2-dimensional (2D) form; we extend this mapping definition f to a 3D case by adding a z-axis into (3):

$$f(x, y, z) = \alpha_0 + \alpha_x x + \alpha_y y + \alpha_z z + \sum_{i=1}^{k} \omega_i \phi(\parallel (x_i, y_i, z_i) - (x, y, z) \parallel) \quad (9)$$

And the boundary conditions in (6) become:

$$\sum_{i=1}^{k} \omega_i x_i = \sum_{i=1}^{k} \omega_i y_i = \sum_{i=1}^{k} \omega_i z_i = 0 \quad (10)$$

By letting $\Phi_{ij} = \phi(\parallel (x_i, y_i, z_i) - (x_j, y_j, z_j) \parallel)$ and $\upsilon_i = f(x_i, y_i, z_i)$, the variables in (7) are updated to ω_i, $\boldsymbol{\alpha} = [\ \alpha_0 \quad \alpha_x \quad \alpha_y \quad \alpha_z\ ]^T$ and the i^{th} row of $\mathbf{L}$, $\mathbf{L}_i = [\ 1 \quad x_i \quad y_i \quad z_i\ ]$.

If a set of feature points in space maintain an invariant spatial relationship, the spatial correspondence of all points in both spaces can be described using a minimum distortion function called Thin Plate Spline (TPS) warping [40]. It is possible to generate a set of candidate waypoints for the task trajectory using this constraint distortion warping algorithm, by assuming

a given scene-matching algorithm, such as SIFT, can provide a set of corresponding ICPs from both the template and the task. Thus, for each $\mathbf{m}_l$ in the template, there exists $q \geq 0$ mapped coordinates as possible candidates for $\mathbf{m'}_l$ in the task defined by (4). This set of candidate waypoints are then subject to *Minimum-Energy Route Plan* to generate a task trajectory.

4.1.3. Minimum-Energy Path Planning

Given the time-series candidate waypoints $\mathbf{m'}$, the goal-directed action can be represented simply by stepping through the variable l and choosing the best point from each $\mathbf{m'}_l$ to form the trajectory. As we use energy consumption as our main selection criterion, the optimisation criterion CE is to minimise the cost function proportional to the square sum of the changes in positions, i.e. translational energy:

$$CE = \sum_{i=2}^{p} (\parallel \mathbf{m'}_i - \mathbf{m'}_{i-1} \parallel)^2 \tag{11}$$

To simplify the computation of the optimisation process, we can define $SD_{ij} = \parallel \mathbf{m'}_i - \mathbf{m'}_j \parallel^2$, in a given time-stage l a possible waypoint as U, $CE_l(U)$ as the minimum energy from the waypoint U to the starting point. Instead of evaluating the full discrete energies mesh through time-steps, we can reduce it to a dynamic programming problem [48]:

$$CE_l(U) = \min_{waypoints\ v\ in\ l-1} \{SD_{uv} + CE_{l-1}(V)\} \tag{12}$$

4.1.4. Iterative Plan Adjustment

In a complex and dynamic environment, the generated trajectory is then checked against the extra invariant feature points F'_x present in the task

Algorithm 4.1: Iterative Plan Adjustment Algorithm (IPA)

Set number of contradiction (N_c) = 0 ;

for *all* j **do**

  Compute $Ct(\mathbf{b}_j)$;

  if $Ct(\mathbf{b}_j) = 1$ **then**

    Find the point $\mathbf{m}'_l$ on the *task* trajectory nearest to $\mathbf{b}_j$;

    Locate the corresponding point $\mathbf{m}_l$ in the *template*;

    Put these $\mathbf{m}_l$ into set F and $\mathbf{b}_j$ into F';

    $N_c = N_c + 1$;

  end

end

if $N_c > 0$ **then**

  Re-iterate the distortion warping algorithm to find a new *task* trajectory;

  Run IPA with the new trajectory and constraints;

end

space, such as forced waypoints or obstacles, i.e. whether any $\mathbf{m}'_l$ contradicts with $\mathbf{b}_j$ in F'_x. For forced waypoints, we can define the contradiction evaluation Ct as

$$Ct(\mathbf{b}_j) = \begin{cases} 1 & \text{if } \mathbf{b}_j \not\subseteq \mathbf{m}' \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

After generating the possible trajectory for the task, the model should perform the algorithm described in Algorithm 4.1 until all $Ct(\mathbf{b}_j) = 0$, i.e.

no contradiction between the set of invariant features and the generated trajectory. It is worthwhile to note that this module alone can be used as a standalone one-shot learning model for trajectory imitation [25].

4.2. Template Segmentation

As it has been hypothesised with biological evidence that an action constitutes a sequence of basic movements [33], we believe that an LbD framework that encodes actions as a sequence of subtasks is useful even from an engineering perspective:

1. It significantly reduces the learning redundancy. For instance, a robot is given demonstrations of a series of grasping actions of a range of objects. Every demonstration comprises of an end-effector approaching an object, grasping it and bring it back. The robot is then asked to imitate all the actions. In the case without action decomposition, we can see that although the kinematics of reaching and retracting actions are likely to be similar for all objects, they are learned as part of grasping a particular object. This redundancy can be minimised if the robot can smartly learn these actions as a series of basic movements and avoid classifying the same subtask as part of a new action.
2. The learned subactions become reusable. A generative model can integrate them to produce new actions without further learning. For instance, an observed action consists of a subaction sequence A-B-C. The robot is then asked to learn a new task that constitutes the sequence C-B-A-A-C. If the robot would have segmented these actions before, it would have realised that no new learning is required for the

kinematics of this action. What it needs to do is to label the new action as a chain-event of these subactions.

3. Segmentation during observation makes the learning generative. In traditional primitive-based learning, primitives are often pre-defined by humans with their intrinsic definitions and knowledge of the basic primitive actions. This set of primitive actions are not generative and not always the most efficient for machine representation. In contrast, segmenting and learning new basic action primitives in perception phase gives more generative flexibility to adopt new primitive actions without subjecting them to human intrinsic knowledge.

To simplify the problem, we will assume that all task actions are composed of simple subactions which are separable spatial-temporally. Therefore, a spatial-temporal clustering algorithm can serve this problem well. In clustering algorithms used for LbD motion segmentation [49], the temporal information is treated as an extra feature dimension. Some undesirable clustering results arisen from treating time and space with the same "resolution" are addressed in [44] by introducing different scaling parameters for time and spatial features in spectral clustering with the Gaussian affinity measure. By setting the scaling parameters to our sampling resolutions for time and space respectively, we can employ this algorithm to segment our templates naturally into clusters of primitive actions.

Once the templates are segmented, the representation among these templates changes from continuous to discrete with symbols representing each segmented subaction. Apart from the original demonstrated task, if the robot is required to perform an action made up of the learned symbols, the issue of

continuity or smooth transition between each action symbol becomes a problem. This appears to be a main hurdle for a number of symbolic LbD models to be useful in action execution. However, one property of the trajectory template adaptation approach makes continuity possible - the set of invariant features. For every segmented template, we make sure that not only the starting position but also the last waypoint in the segmented trajectory are added as invariant feature points.

4.3. Online Template Learning

One-shot learning, to a large extent, is favourable in real-world LbD and HRI because users are not required to repeat the same actions over many trials in a given time frame. Moreover, in some extreme cases, repetition of the same action is not possible. However, a one-shot learning model has a few assumptions/disadvantages:

- As it is similar to a winner-take-all strategy, the underlying assumption is that the demonstration for each action is close to perfect. Once learned, it can only be replaced by a better demonstration of the same action.

- It also assumes that the noise associated with each perceived demonstration is minimal so that the learned action can be executed without using a generalisation technique to remove noise over repeated trials.

- It is further assumed that the demonstrated action does not evolve over time. Or every improvement can be evaluated by a metric, and hence replaces the previous template.

The direct application of a template based approach suffers from the same problem as one-shot learning paradigms - only the best-fit trajectory gets stored as a template. This drawback becomes prominent when all demonstrated actions for a particular template are noisy. The noise gets also stored into the template due to the lack of generalisation capability over multiple training samples. However, treatment using an online learning technique [43] can be applied to address this problem. When multiple demonstrations for a template are given, the template can be updated each at a time to produce a more generalised template.

To allow continuous learning and generalising a particular template, we propose to insert an online learning (OL) module into the framework. The chosen OL model has to possess the flexibility to adapt to a wide range of dynamic demonstrations as many trajectory demonstrations may be highly non-linear. Thus, non-parametric approaches are preferred as tuning of a generative model can be time-consuming and demand the users to possess the expertise. It is also preferred for an OL model to provide a confidence estimate so that we can evaluate if generalising a particular template is suitable by setting a confidence threshold for the update.

The Online Echo-State Gaussian Process (OESGP) illustrated in Figure 2, to our best knowledge, is the only model that possesses all the qualities we require. This sparse Bayesian formulation of echo-state network enables fast iterative learning with uncertainty feedback. In our application, we use the available OESGP implementation in a two-stage manner:

Stage 1 : When a new template is presented, it gets recorded directly into the database while an OESGP model is trained to represent the action.

However, the output from the online learning does not overwrite the recorded template.

Stage 2 : When a new demonstration of an existing template is presented, the associated OESGP model is updated with the new demonstration. The mean output of the model then overwrites the template in the database.

4.4. Primitive Decomposition

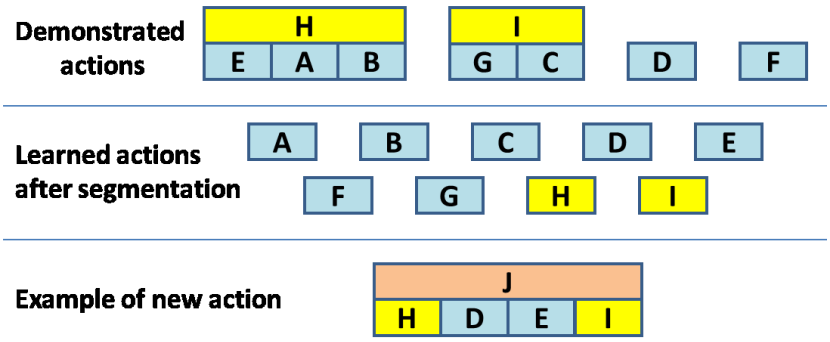


Figure 5: An example of primitive decomposition. Assuming a robot learns the primitive actions D and F and the compound actions H and I in 4 demonstrations. The template segmentation module thus segments H and I into a list of subactions. The template database now has all primitives from A to G which can be used to construct a new action J.

Another difficulty in direct use of templates lies in the partial reuse of the learned actions to form a new action, thereby reducing the learning redundancy. Thus, the learned action template needs to be segmented into a sequence of basic movement primitives [33] and expressed as a chain-event of subactions similar to the hierarchical action control structure in human cortex [34].

Koechlin and Jubault [34] suggest that a hierarchical control structure exists in the human brain which executes actions at the levels of superordinate chunks, simple chunks and single acts. Since an action has been segmented into a series of primitive templates and tagged, a new action can therefore be formed by integrating multiple learned actions and primitive templates as illustrated in Figure 5. For example, given an unseen action J which is composed of learned subactions H-D-E-I, the robot can perform this action by using a simple finite state machine (FSM) to piece the subactions up. This can be achieved in two ways:

- The user manually creates the FSM with a relevant set of invariant constraint features in the task space hard-coded into each of the subtask or extracted from the scene of the task.

- The user can also demonstrate the new action J before the robot, this allows the robot to extract the invariant features from the perceived environment and also segment the action. Once segmented, the subactions are compared against the database using the constraint distortion warping method, in which the energy measure (1) can be used to evaluate the template similarity with a threshold value. It then labels the unseen subactions, if any, and composes a FSM to represent action J.

We will demonstrate the first case in Section 6.1 with the implementation from our previous work in [50] and the second case in Section 6.2.

5. Statistical Experiments

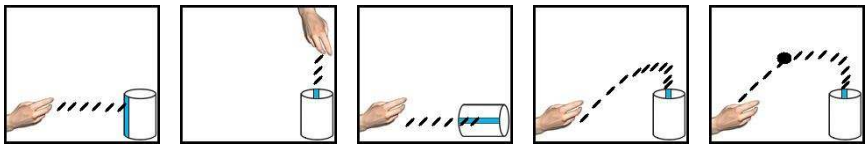
In this section, we report on a series of experiments designed to statistically investigate the relative performance of each module using the same benchmark dataset.

5.1. The Benchmark Dataset

(a) (b) (c)

Figure 6: The experiment set-up for collection of the benchmark dataset. (a) The 53 Degrees-of-Freedom open-sourced iCub robot is developed by the RobotCub Consortium [51, 52]. (b) and (c) are an example of the human subject with colour markers captured by the iCub cameras.

We use the on-board cameras of the iCub humanoid robot for the experiments. All demonstrations are performed in front of the iCub cameras with a fixed configuration. Throughout data collection, the stereo cameras of the iCub were set at 20Hz frame rate and 320 X 240 pixel resolution (example shown in Figure 6b & 6c). To avoid the tracking issues in computer vision at this resolution, a number of colour trackers are placed along the hand of each human subject. The recorded demonstrations are processed offline using the standard OpenCV blob tracking.



(a) (b) (c) (d) (e)

Figure 7: The sketches of the 5 sets of tasks. The hand indicates the starting point for each task. The blue strip denotes the approach plane that the hand approach has to be perpendicular to which is indicated by an array normal to the strip marked on the cover of the cylinder.

In our experiment, we consider a real-world task of grasp-oriented reaching actions. To benchmark the robustness of the proposed framework, a total of five different experimental tasks are carried out as described below and illustrated in Figure 7:

1. Task 1 (Figure 7a): The demonstrator is to approach a standing can from the left of the scene; the approach plane, denoted as a blue strip, is perpendicular to the direction of motion. The whole movement is expected to be planar.
2. Task 2 (Figure 7b): The demonstrator is to approach the object from the far side of the scene. The only difference to Experiment 1 is the whole set-up is rotated 90° clockwise.
3. Task 3 (Figure 7c): The demonstrator is to approach the object from the left, however, the approach plane is facing up. The movement is expected to gradually become 3-dimensional when the hand approaches the object.
4. Task 4 (Figure 7d): The demonstrator is to approach the object from the left, however, the approach plane is facing the far side.

5. Task 5 (Figure 7e): The set-up is similar to that of Experiment 4 with addition of an extra waypoint, denoted by the black patch, that the demonstrator has to navigate through.

Each task consists of 15 trials performed by different subjects. All subjects are requested to perform the demonstrations as naturally as possible. And we extract the action trajectories from all trials and attach the relevant set of environment-invariant features to them. This set of constraint features includes the start position of the hand and the corresponding corners of the bounding box and centroid of cylinder.

5.2. Experiment Descriptions

The 75 collected human trials are firstly labelled and grouped according to their original task numbers before we carry out the following statistical experiments:

1. Take the constraint features of each of the 75 trials as both template and task constraints, run the template adaptation module to obtain a set of 75 self-mapped trajectories. The output and the demonstrated trajectories are compared according to their task numbers to estimate the amount of distortion introduced by the model.
2. Take the constraint features of each of the 75 trials as the template constraints, and map them to the constraints of all trials to obtain a 75×75 set of generated trajectories using only the template adaptation module. The output and the demonstrated trajectories are grouped in each of the input/output pair for quantitative analysis of how well the model can generalise templates to different scenarios.

3. Take each of the 75 trials as input, run the online template learning module to generate outputs for other trials within the same task. This generates $5 \times 15 \times 15$ pairs of trajectories for benchmarking analysis against the performance of two other LbD models.
4. For each task, we carry out 2 further experiments: a) Train the model with 14 input trajectories and test the model on the last trajectory; b) Use all 15 trials as both training and testing samples. The statistical difference between the standard leave-one-out training and trained all will provide an indication of possible model over-fitting.
5. Uniformly segmented in to $N \in 2 \dots 4$ segments in time after applying dynamic time warping (DTW) [53]. We hypothesise that the decomposition of actions into smaller templates helps to improve the learning outcome, as the constraint features are broken down specifically and locally for each subaction. We will perform mesh-mapping for all 75 trials with different levels of segmentation using template adaptation to generate the corresponding trajectories. This yields a $75 \times 75 \times 4$ tensor of trajectories for cross-validation to test for improvement of performance after template segmentation.

5.3. Statistical Evaluation Metrics

We introduce two performance metrics to statistically evaluate the performance of the framework - Correlation Coefficient and Mean Squared Difference. We denote the generated trajectory as $\mathbf{m}$ and that to be compared as $\mathbf{m}'$. Both consist of N corresponding waypoints after DTW.

Assuming that the proposed algorithm could estimate the resulted trajectory performed by human under similar circumstances, the Correlation

Coefficient R is a likeliness indicator of our proposed framework to predict the human demonstrated trajectories.

$$R = \frac{\sum_{i=1}^{N}(\mathbf{m}_i - \bar{\mathbf{m}}) \cdot (\mathbf{m}'_i - \bar{\mathbf{m}}')}{\sqrt{(\sum_{i=1}^{N}(\mathbf{m}_i - \bar{\mathbf{m}})^2)(\sum_{i=1}^{N}(\mathbf{m}'_i - \bar{\mathbf{m}}')^2)}} \tag{14}$$

where $\bar{\mathbf{m}}$ denotes the arithmetic mean of $\mathbf{m}_i$.

We also use the Mean Squared Differenc (MSD) to estimate the closeness between the generated trajectories and the human demonstrations.

$$MSD = \frac{1}{N}\sum_{i=1}^{N} \parallel \mathbf{m}_i - \mathbf{m}'_i \parallel^2 \tag{15}$$

5.4. Results and Discussion

Table 1: Performances of the template adaptation module for all self-mapping cases

Indicators	Tsk 1	Tsk 2	Tsk 3	Tsk 4	Tsk 5
R	0.996	0.999	0.991	0.991	0.994
MSD	24.6	10.7	41.2	38.6	27.0

First, the performance of the template adaptation module on the ability to preserve the original trajectory before and after mapping was evaluated. This can be measured by the performance of applying the same input and output constraints associated with a particular demonstration. We tabulate them according to the experiments they belong to as shown in TABLE 1. All five experiments have correlation coefficients greater than 99% and very low MSDs, which suggest that the template adaptation module do not impose distortion to the original signals.

Table 2: Mean R/MSD for mapping from one task to another. Columns and rows indicate input and output respectively.

	Tsk 1	Tsk 2	Tsk 3	Tsk 4	Tsk 5
Tsk 1	0.961/176	0.953/298	0.817/1222	0.414/7388	0.380/8390
Tsk 2	0.993/117	0.995/57	0.941/1899	0.740/2809	0.711/5095
Tsk 3	0.861/799	0.444/2832	0.891/399	0.774/962	0.747/2869
Tsk 4	0.757/1548	0.817/667	0.871/1164	0.957/203	0.938/343
Tsk 5	0.835/1139	0.816/834	0.872/1286	0.885/459	0.962/280

We grouped the mean of the performance indicators according to each input/output task to tabulate how well the trajectories generated from demonstrations of a particular task can be generalised into other task scenarios with the template adaptation module. The results are shown in TABLE 2. The R statistics suggest that the proposed module is capable of generating trajectories that are very close to those performed by humans as 88% of the R statistics are above 0.7. It also indicates that mapping a less complex route, e.g. a straight line or the trajectories without forced waypoints, to any given scenario results in performance closer to those of humans. The likely reason for such observation is the lack of corresponding extra invariant feature information in the target situation for mapping the complex case to simpler ones which can result in poor performance. It can also be inferred that mapping demonstrations of similar scenarios, for example Task 4 and Task 5, may produce similar results even when some defining conditions are different. We can further see that mapping trajectories from Tasks 1-4 to Task 5 yields

good performance which implies that the IPA algorithm works sensibly well in these situations. The above results indicate that the template adaptation algorithm can, while maintaining good adaptability, reduce the burden and cost of repeated demonstrations; thus our aim to address this issue in LbD can be met by the Template Adaptation Module [50].

Table 3: Mean MSD using online template learning module (using OESGP) against the benchmark results of other ESN models. ESN and CESN denote echo-state network and Copula echo-state network respectively.

MSD	Task 1	Task 2	Task 3	Task 4	Task 5
ESN [54]	1447	914	1166	8212	721
CESN [55]	1097	150	787	601	627
OESGP	613	45	301	151	120

We then targeted the online template learning module (OTL). In all our experiments, the OESGP in the OTL module is implemented with noise parameter set to 0.1 and memory size of 20 which is significantly lower than the length of any demonstration. The memory size here refers to the maximum number of the reservoir states retained at each OESGP iteration. With this set-up, the trained model using any demonstration as both training and testing sample yield $R = 1.000$. Thus, we further evaluate the generalisation ability of the model by training it with one demonstration from an experiment and testing on the other 14 trials. The results are then compared against the results in the literature using the same dataset as shown in TABLE 3. The table shows that the OTL module performs better than the

ESN models proposed in [54, 55] with approximately doubled improvement over the figures in MSD.

Table 4: Mean R/MSD for the 3 experiments conducted on the online template learning module.

Tasks	1	2	3	4	5
Train 1	0.956/613	0.993/45	0.951/301	0.970/151	0.981/120
Train 14	0.9997/2.9	0.9998/0.8	0.9995/3.8	0.9994/2.6	0.9996/2.1
Train 15	0.9999/1.2	0.9998/0.5	0.9997/1.4	0.9996/1.4	0.9997/1.3

To further evaluate the generalisation capability of the OTL module, the results of the two further sets of experiments are tabulated in TABLE 4. The R statistics is well above 90% positively correlated right from only 1 training sample. This suggests that the dynamics of the demonstrated data can be well explained by the model. Moreover, no significant improvement by increasing the training samples from 14 to 15, which is the full set. There may be an issue of over-fitting for supplying all samples as both training and testing data. However, the high R statistics right from training with 1 example together with insignificant improvement by increasing the training from 14 to 15 samples suggests that the model is unlikely over-fitted.

We tabulated the R and MSD statistics for the number of segments we introduced to a learned template. TABLE 5 shows the R and MSD statistics for the 75 self-mapping trials arranged by the number of segments. We can see that in all cases, the correlation coefficient is almost 100% with extremely low MSD. The statistics is improved as soon as segmentation is introduced.

Table 5: Mean R/MSD of self-mapping cases with different No. of segments

Segments	Task 1	Task 2	Task 3	Task 4	Task 5
1	0.996/24.6	0.999/10.7	0.991/41.2	0.991/38.6	0.994/27.0
2	0.999/12.1	1.000/7.2	0.999/11.8	1.000/6.6	1.000/6.9
3	0.999/11.8	1.000/6.7	0.999/11.1	1.000/6.2	1.000/6.5
4	0.999/10.3	1.000/6.5	0.999/10.9	1.000/5.9	1.000/6.0

Two inferences can be drawn from this observation: 1) The system does not introduce significant distortion to the original signals with and without the template segmentation module; 2) The introduction of template segmentation module, may help to reduce noise in the mapping by preserving only the local constraint features which as a consequence helps to preserve the original signal further.

TABLE 6 shows the statistics of the mappings with the template adaptation module for the uniformly cut segments. As previously mentioned, we hypothesised that the observation of poor performance for mapping from complex cases to simpler ones is expected due to the lack of corresponding extra feature points. This is confirmed by the statistics for the segmented templates as the R surge to more than 80% as soon as template segmentation was introduced. When the number of segment reaches four, almost all R statistics are above 0.95.

The above results indicate that our previous claim can be sustained while maintaining good adaptability the template adaptation algorithm reduces the need of repeated demonstrations [56]. Moreover, our template segmenta-

Table 6: Mean R/MSD for mapping from one task (1-4 segments) to another.

	Tsk 1	Tsk 2	Tsk 3	Tsk 4	Tsk 5
Tsk 1	0.961/176	0.953/298	0.817/1222	0.414/7388	0.380/8390
	0.979/74	0.941/177	0.860/682	0.800/538	0.791/562
	0.987/31	0.976/63	0.948/133	0.949/120	0.934/156
	0.989/18	0.984/30	0.977/58	0.978/55	0.966/76
Tsk 2	0.993/117	0.995/57	0.941/1899	0.740/2809	0.711/5095
	0.970/436	0.998/26	0.877/2448	0.972/378	0.964/444
	0.989/152	0.999/8	0.970/356	0.994/78	0.990/136
	0.995/70	0.999/4	0.988/122	0.997/28	0.995/56
Tsk 3	0.861/799	0.444/2832	0.891/399	0.774/962	0.747/2869
	0.910/385	0.896/316	0.941/231	0.877/619	0.847/852
	0.952/143	0.948/117	0.967/122	0.963/110	0.954/140
	0.966/70	0.968/55	0.982/62	0.980/57	0.973/68
Tsk 4	0.757/1548	0.817/667	0.871/1164	0.957/203	0.938/343
	0.872/606	0.845/551	0.798/2317	0.977/148	0.967/176
	0.929/211	0.922/219	0.938/397	0.987/69	0.981/94
	0.944/115	0.954/111	0.969/175	0.992/41	0.987/59
Tsk 5	0.835/1139	0.816/834	0.872/1286	0.885/459	0.962/280
	0.875/758	0.879/478	0.840/1963	0.966/222	0.977/133
	0.937/255	0.935/261	0.935/428	0.979/124	0.987/75
	0.952/151	0.960/172	0.965/215	0.984/92	0.991/44

tion approach can help introduce further invariant features into the template and make the adaptation more accurate while the online template learning module maintains model generalisation capability.

6. Real-World Applications

6.1. The Tic-tac-toe Game

In this section, we present a demonstration of robot playing the tic-tac-toe game with only one demonstration. It demonstrates the relative merits of our framework in teaching the moves to a robot in LbD. The iCub humanoid robot is shown one demonstration of how to move in the grid of the tic-tac-toe game sheet and place a mark of a given shape illustrated in Figure 8.

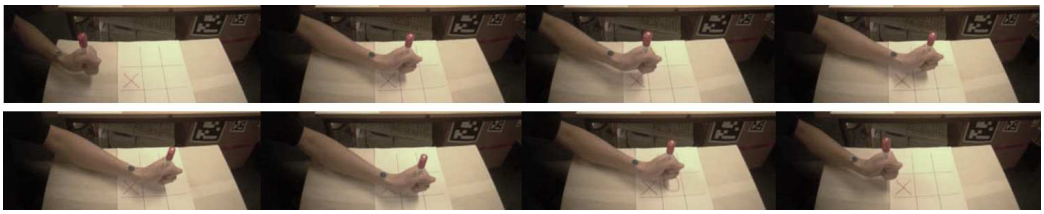


Figure 8: The iCub camera view of demonstration of placing a circle in the grid-space of the tic-tac-toe game.

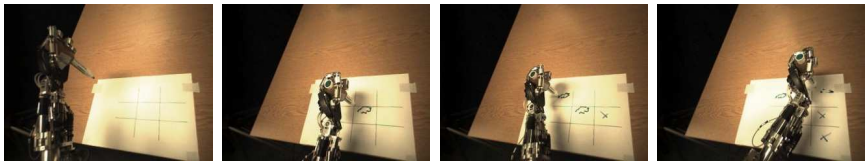
Although the human subject is asked to perform a planar movement as much as possible, the arm of the iCub is randomly parked outside and above the grid space. Moreover, the iCub is required to play the game in a new tic-tac-toe grid with different location, size and orientation.

In this game, we assumed that the marker to place the mark was always in the hand of the iCub and the invariant features can be fully extracted from the sheet of paper containing the grid space. We demonstrated two

applications of this game - without and with template segmentation of the demonstrated movement. In the set-up without template segmentation, the iCub hand always returned to its initial position as it follows the entire demonstration. In the latter, the iCub hand would only return to its home position when it finishes playing the game. As the template actions were recorded in Cartesian space, we used the on-board inverse kinematics of the iCub to execute all the actions.

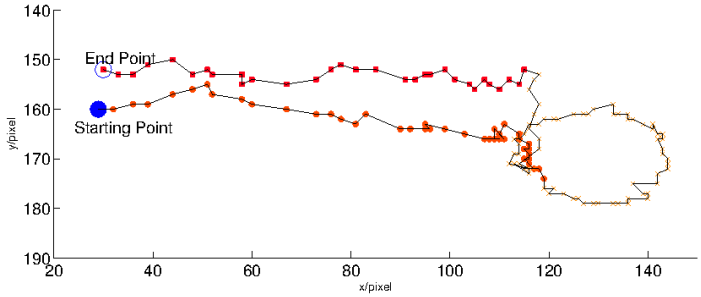
Figure 9 shows snapshots of the iCub playing the game. The difference between the two applications are denoted by the plots in Figures 10b and 10c. The demonstration is naturally divided into three segments by the template segmentation module, as illustrated in Figure 10a. These segments correspond to "reach", "act" and "withdraw" actions. We therefore implemented an FSM for the primitive decomposition module to control the iCub to finish playing the game before returning the arm to its home position.

(a)

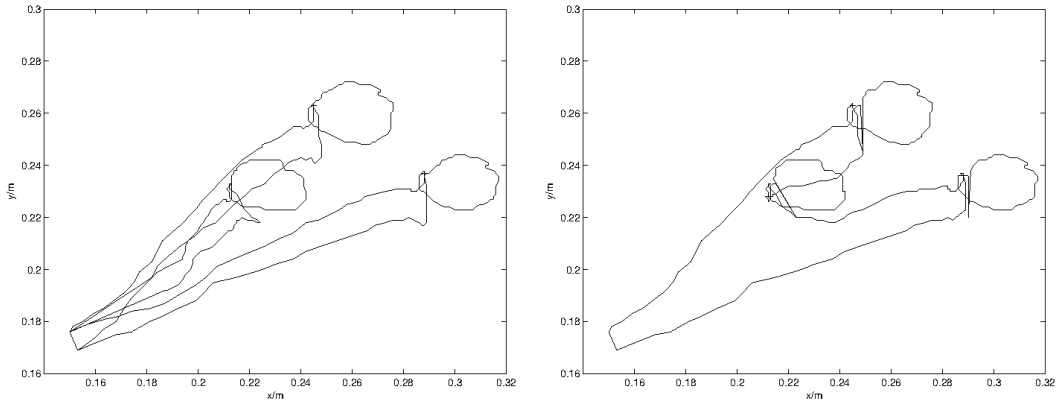


(b)

Figure 9: iCub playing tic-tac-toe game without (a) and with (b) template segmentation. In (a), the iCub returned its arm to the parking position after completing its turn while in (b) the iCub returned its arm only after completing the entire game.



(a)



(b) (c)

Figure 10: Plots of the demonstrated and imitated movements. (a) shows the demonstrated trajectory in the iCub-camera view segmented by the Template Segmentation module. (b) and (c) show the generated trajectories for the game without and with Template Segmentation.

The correlation coefficients of the all the circles generated by the framework are above 0.9. Figures 10b and 10c. Figures 10b and 10c show the difference between the games played without and with the Template Segmentation Module. When the Template Segmentation Module was in use, the iCub removed the redundancy having to constantly move between the park-

ing position and the desired cells. From Figure 9, we can see that the iCub marked all the symbols in the right cells with fair accuracy. The iCub drawn circles are much more jerky compared to those generated by the framework shown in Figures 10b and 10c. This may be due to the control commands executed by the iCub's inverse kinematics module which do not try to move across two points using the shortest route. However, the discrimination between the 2 different symbols in the game is clearly not affected by this.

6.2. Soldering Task

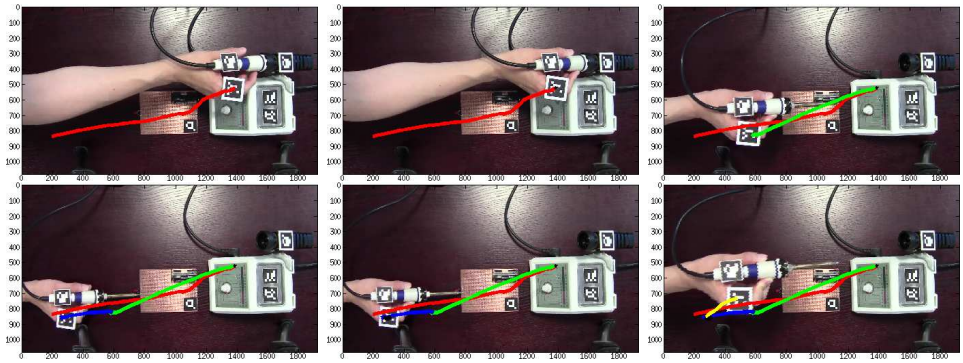
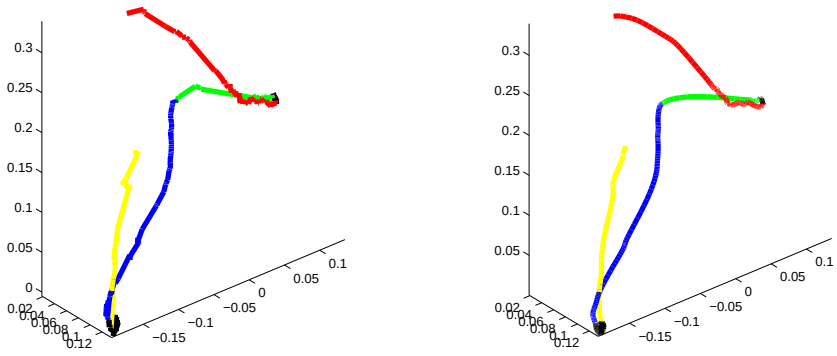


Figure 11: Human demonstration of the soldering task. Each scene with a new coloured trajectory represents a segmented phase of action by the template segmentation module.

We showcase the full framework on the task of robot soldering a simple circuit board learned in an LbD setting. The human demonstrates the entire sequence of picking up a soldering iron and applying it at the circuit board. The sequence is illustrated in Figure 11. Two demonstrations of the task were used to train the framework as illustrated in Figure 12.

Both the scene objects and the human hand have been marked by the 2D AR Markers for accurate tracking, convenient location of invariant features and to minimise correspondence issues. Figure 12 shows the extracted



(a) (b)

Figure 12: The 3D plot of the template soldering trajectories. The coloured segments correspond to the trajectories illustrated in Figure 11. (a) is the plot of the templates after 1 demonstration while (b) is the plot after 2 demonstrations.

and segmented primitive templates in a 3D plot after 1 and 2 training examples . Each coloured segment represents a primitive template segmented by the framework. They are “approach tool”, ”grasping”, ”withdraw tool”, ”approach target”, ”soldering” and ”leave target” respectively.

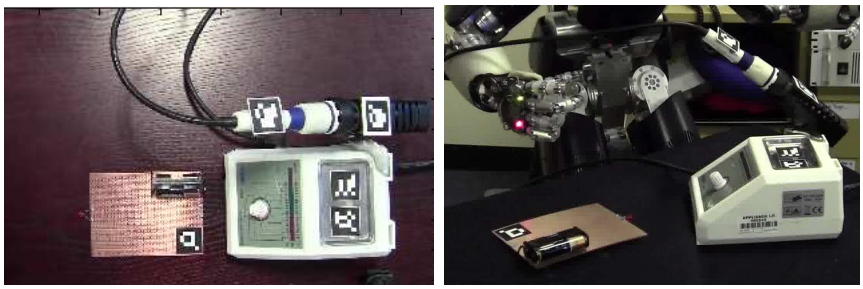


Figure 13: The scenes where the soldering task are performed by the human (left) and the iCub (right).

The robot is then given the same soldering task, however, with the location and pose of the scene objects changed as illustrated in Figure 13. As the gross movement of the end-effector during grasping and soldering is infinitesimal and the hand of the iCub is under-actuated, dexterous manipulation and grasping tasks with this hand which has to readily react to rapid environmental changes [57] are a separate topic beyond LbD [58, 59]. Thus, the templates for these two actions are supplied to the grasping model proposed in [58] and the manipulation model proposed [59] for analysis and action execution in the primitive decomposition module. Furthermore, to reduce the jerky controller response, we increased the template recording frequency by using an external camera with $40Hz$ frame-rate to capture the demonstration. This allowed us to send continuous commands of small displacements to the controller without having to wait for the execution to finish.

Table 7: Statistics of the learned template kinematics

After n^{th} Demo	Speed($\mu m/s$)	Acceleration($\mu m/s^{-2}$)	Jerk($\mu m/s^{-3}$)
n=1	2986±7830	3248±10253	2853±9852
n=2	1697±3013	241±531	94±196

To analyse the difference between the templates after the first and second demonstrations, we calculated the trajectory difference and tabulated the kinematic statistics for each template in TABLE 7. The mean path difference of the templates before and after the second demonstration is $0.0029\pm0.0057m$. Although this difference is negligible, the 1^{st}, 2^{nd} and 3^{rd} derivatives between the template trajectories are in different orders of

magnitude. This can be confirmed from the plots in Figure 12. The template recorded after the first demonstration is the perceived demonstration itself. Due to visual localisation errors of the AR markers, noise is present in every perceived demonstration. This is represented by the high path jerkiness. This problem can either be treated using a filtering technique on the current trajectory with some noise estimate or a learning technique over repeated demonstration of the task. Thus, after the 2^{nd} demonstration, the online template learning module generalised the demonstrations to produce a "cleaner" template trajectory as the jerk after the 2^{nd} dropped by two orders of magnitude.

The soldering task executed by the iCub is shown in Figure 14. The sequence was executed using the iCub inverse-kinematics module. It can be clearly identified that the LED light was been lit after the soldering task had completed. A video of this demo can be found at http://www.imperial.ac.uk/personalrobotics/yanrasvideos.

7. Conclusion

In this paper, we presented a morphable primitive template-based learning framework to address the "how-to" problem in robot learning by demonstration. The proposed framework is a mapping function approach used at trajectory learning level with the incorporation of some relative merits of system models. It decomposes a demonstration into a series of primitive templates with recordings of their associate constraints. Multiple demonstrations of the same templates are treated using an online learning method for generalisation. We extended the Thin-Plate Spline Warping algorithm for

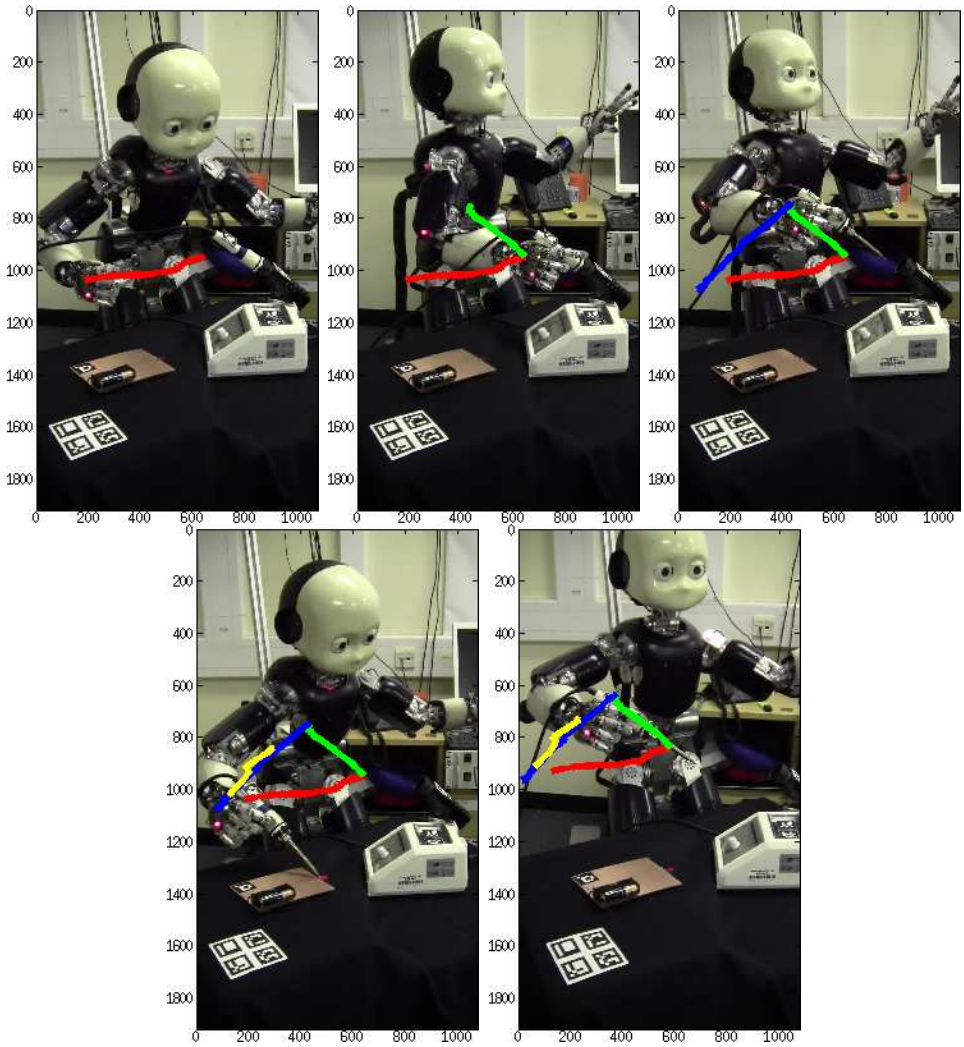


Figure 14: The iCub performing the soldering task. The coloured segments correspond to the trajectories illustrated in Figure 11.

3D trajectory adaptation in order to generate actions in a new environment with different constraint locations.

Our proposed framework was statistically evaluated using a set of benchmark experiments. All modules were evaluated to show statistically that the framework components have the flexibility to adapt to situations, are able to generalise well and outperformed some benchmarked approaches. We also implemented the framework on the iCub humanoid robot to play a real-life tic-tac-toe game and perform a soldering task. As mentioned earlier on, the current framework applying on an under-actuated hand relies on external control-based models to aid the execution. We plan to extend the framework further to incorporate imitation learning at dynamics level.

8. Acknowledgements

This work was supported in part by the EU FP7 project WYSIWYD under Grant 612139. The authors gratefully acknowledge the invaluable comments and continued support from members and alumni of the Imperial College Personal Robotics Lab.

References

[1] A. Billard, S. Calinon, R. Dillmann, S. Schaal, B. Siciliano, O. Khatib, Robot Programming by Demonstration, in: B. Siciliano, O. Khatib (Eds.), Springer Handbook of Robotics, Springer Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 1371–1394.

[2] B. D. Argall, S. Chernova, M. Veloso, B. Browning, A Survey of Robot

Learning from Demonstration, Robotics and Autonomous Systems 57 (2009) 469–483.

[3] Y. Demiris, L. Aziz-Zadeh, J. Bonaiuto, Information processing in the mirror neuron system in primates and machines, Neuroinformatics 12 (2014) 63–91.

[4] A. Alissandrakis, C. L. Nehaniv, K. Dautenhahn, Imitation with ALICE: Learning to Imitate Corresponding Actions Across Dissimilar Embodiments, IEEE Transactions on Systems, Man and Cybernetics, Part A 32 (2002) 482–496.

[5] S. Munch, J. Kreuziger, M. Kaiser, R. Dillmann, Robot Programming by Demonstration (RPD) - Using Machine Learning and User Interaction Methods for the Development of Easy and Comfortable Robot Programming Systems, in: Proceedings of the 24th International Symposium on Industrial Robots, volume 25, JIRA, 1994, pp. 685 – 693.

[6] J. Peters, S. Vijayakumar, S. Schaal, Reinforcement Learning for Humanoid Robotics, in: Proceedings of the 3rd IEEE-RAS International Conference on Humanoid Robots (Humanoids), IEEE, Karlsruhe, Germany, 2003, pp. 1–20.

[7] Y. Yoshikawa, K. Shinozawa, H. Ishiguro, N. Hagita, T. Miyamoto, Responsive Robot Gaze to Interaction Partner, in: Proceedings of the 2006 Robotics: Science and Systems Conference (RSS), pp. 287–293.

[8] S. Schaal, A. Ijspeert, A. Billard, Computational Approaches to Motor

Learning by Imitation, Philosophical Transactions of the Royal Society of London. Series B: Biological Sciences 358 (2003) 537–547.

[9] S. Calinon, F. Guenter, A. Billard, On Learning the Statistical Representation of a Task and Generalizing it to Various Contexts, in: Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2006, pp. 2978–2983.

[10] K. Dautenhahn, Roles and Functions of Robots in Human Society-Implications from Research in Autism Therapy, Robotica 21 (2003) 443–452.

[11] K. Dautenhahn, C. L. Nehaniv, The correspondence problem, MIT Press, 2002.

[12] D. G. Lowe, Object Recognition from Local Scale-Invariant Features, in: Proceedings of the 7th IEEE International Conference on Computer Vision (ICCV), volume 2, IEEE, 1999, pp. 1150–1157.

[13] S. P. Chatzis, D. Korkinof, Y. Demiris, A Nonparametric Bayesian Approach toward Robot Learning by Demonstration, Robotics and Autonomous Systems 60 (2012) 789–802.

[14] Z. Ghahramani, M. I. Jordan, Supervised Learning from Incomplete Data via an EM Approach, in: J. D. Cowan, G. Tesauro, J. Alspector (Eds.), Advances in Neural Information Processing Systems, volume 6, Morgan Kaufmann Publishers, Inc., 1994, pp. 120–127.

[15] S. M. Khansari-Zadeh, A. Billard, Learning Stable Nonlinear Dynamical Systems With Gaussian Mixture Models, IEEE Transactions on Robotics 27 (2011) 943–957.

[16] A. Billard, S. Calinon, F. Guenter, Discriminative and Adaptive Imitation in Uni-manual and Bi-manual Tasks, Robotics and Autonomous Systems 54 (2006) 370–384.

[17] J. Demiris, G. M. Hayes, Imitation as a Dual-route Process Featuring Predictive and Learning Components: a Biologically Plausible Computational Model, in: Imitation in Animals and Artifacts, MIT Press, Cambridge, MA, USA, 2002, pp. 327–361.

[18] A. Ude, C. G. Atkeson, M. Riley, Programming Full-body Movements for Humanoid Robots by Observation, Robotics and Autonomous Systems 47 (2004) 93–108.

[19] R. Dillmann, Teaching and Learning of Robot Tasks via Observation of Human Performance, Robotics and Autonomous Systems 47 (2004) 109–116.

[20] V. Mohan, G. Metta, J. Zenzeri, P. Morasso, Teaching Humanoids to Imitate Shapes of Movements, in: K. Diamantaras, W. Duch, L. Iliadis (Eds.), Proceedings of the 2010 International Conference on Artificial Neural Networks (ICANN), volume 6353, Springer Berlin Heidelberg, 2010, pp. 234–244.

[21] B. Akgun, M. Cakmak, J. W. Yoo, A. L. Thomaz, Trajectories and Keyframes for Kinesthetic Teaching, in: Proceedings of the seventh an-

nual ACM/IEEE international conference on Human-Robot Interaction (HRI), ACM Press, New York, New York, USA, 2012, pp. 391–398.

[22] S. Chernova, M. Veloso, Interactive policy learning through confidence-based autonomy, Journal of Artificial Intelligence Research 34 (2009) 1.

[23] Michael C. Burl, M. Weber, P. Perona, A Probabilistic Approach to Object Recognition Using Local Photometry and Global Geometry, in: H. Burkhardt, B. Neumann (Eds.), Proceedings of the 5th European Conference on Computer Vision (ECCV), volume II of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin/Heidelberg, 1998, pp. 628–641.

[24] M. Lagarde, P. Andry, P. Gaussier, The Role of Internal Oscillators for the One-Shot Learning of Complex Temporal Sequences, in: J. Marques de Sa, L. A. Alexandre, W. Duch, D. Mandic (Eds.), Proceedings of the 17th International Conference on Artificial Neural Networks (ICANN), Lecture Notes in Computer Science, Springer Berlin Heidelberg, Porto, Portugal, 2007, pp. 934–943.

[25] Y. Wu, Y. Demiris, Towards One Shot Learning by Imitation for Humanoid Robots, in: Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA), IEEE, Anchorage, United States, 2010, pp. 2889–2894.

[26] T. M. Mitchell, R. M. Keller, S. T. Kedar-Cabelli, Explanation-based Generalization: A Unifying View, Machine Learning 1 (1986) 47–80.

[27] H. Friedrich, R. Dillmann, O. Rogalla, H. Christensen, H. Bunke, H. Noltemeier, Interactive Robot Programming based on Human Demonstration and Advice, in: H. I. Christensen, H. Bunke, H. Noltemeier (Eds.), Sensor Based Intelligent Robots, volume 1724 of *Lecture Notes in Computer Science*, Springer, Berlin, Heidelberg, 1999, pp. 96–119.

[28] S. Calinon, A. Billard, Stochastic Gesture Production and Recognition Model for a Humanoid Robot, in: Proceedings of the 2004 IEEE-RSJ International Conference on Intelligent Robots and Systems (IROS), volume 3, IEEE, 2004, pp. 2769–2774.

[29] S. Calinon, F. Guenter, A. Billard, Goal-Directed Imitation in a Humanoid Robot, in: Proceedings of the 2005 IEEE International Conference on Robotics and Automation (ICRA), April, IEEE, 2005, pp. 299–304.

[30] D. B. Grimes, R. Chalodhorn, R. P. N. Rao, Dynamic Imitation in a Humanoid Robot through Nonparametric Probabilistic Inference, in: Proceedings of the 2006 Robotics: Science and Systems Conference (RSS), MIT Press, Philadelphia, United States, 2006, pp. 199–206.

[31] S. Schaal, Is Imitation Learning the Route to Humanoid Robots?, Trends in Cognitive Sciences 3 (1999) 233–242.

[32] Y. Demiris, B. Khadhouri, Hierarchical Attentive Multiple Models for Execution and Recognition of Actions, Robotics and Autonomous Systems 54 (2006) 361–369.

[33] J. F. Allen, Towards a General Theory of Action and Time, Artificial Intelligence 23 (1984) 123–154.

[34] E. Koechlin, T. Jubault, Broca's Area and the Hierarchical Organization of Human Behavior, Neuron 50 (2006) 963–74.

[35] D. Kulic, C. Ott, D. Lee, J. Ishikawa, Y. Nakamura, Incremental Learning of Full Body Motion Primitives and Their Sequencing through Human Motion Observation, The International Journal of Robotics Research 31 (2011) 330–345.

[36] G. Konidaris, S. Kuindersma, R. Grupen, A. Barto, Robot learning from demonstration by constructing skill trees, The International Journal of Robotics Research 31 (2012) 360–375.

[37] O. Mangin, P.-Y. Oudeyer, Learning to Recognize Parallel Combinations of Human Motion Primitives with Linguistic Descriptions Using Non-negative Matrix Factorization, in: Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, 2012, pp. 3268–3275.

[38] L. Pais, K. Umezawa, Y. Nakamura, A. Billard, Learning Robot Skills Through Motion Segmentation and Constraints Extraction, in: Workshop on Collaborative Manipulation, ACM/IEEE International Conference on Human Robot Interaction, ACM Press, Tokyo, Japan, 2013.

[39] H. Chui, Non-rigid Point Matching: Algorithms, Extensions and Applications, Ph.D. thesis, Yale University, 2001.

[40] H. Chui, A. Rangarajan, A New Point Matching Algorithm for Non-Rigid Registration, Computer Vision and Image Understanding 89 (2003) 114–141.

[41] B. Takacs, Y. Demiris, Multi-robot Plan Adaptation by Constrained Minimal Distortion Feature Mapping, in: Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2009, pp. 742–749.

[42] F. L. Bookstein, Principal warps: Thin-plate Splines and the Decomposition of Deformations, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (1989) 567–585.

[43] H. Soh, Y. Demiris, Iterative Temporal Learning and Prediction with the Sparse Online Echo State Gaussian Process, in: Proceedings of the 2012 International Joint Conference on Neural Networks (IJCNN).

[44] B. Takacs, Y. Demiris, Balancing Spectral Clustering for Segmenting Spatio-temporal Observations of Multi-agent Systems, in: Proceedings of the 8th IEEE International Conference on Data Mining (ICDM), IEEE, Los Alamitos, CA, USA, 2008, pp. 580–587.

[45] M. J. D. Powell, A Thin Plate Spline Method for Mapping Curves into Curves in Two Dimensions, in: Proceedings of the 1995 Computational Techniques and Applications Conference (CTAC), University of Cambridge Press, Cambridge, 1995.

[46] J. Zhu, M. R. Lyu, Progressive Finite Newton Approach To Real-time Nonrigid Surface Detection, in: Proceedings of the 2007 IEEE Con-

ference on Computer Vision and Pattern Recognition (CVPR), IEEE, 2007, pp. 1–8.

[47] J. Pilet, V. Lepetit, P. Fua, Real-time Non-Rigid Surface Detection, in: Proceedings of the 2005 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), volume 1, IEEE, 2005, pp. 822–828.

[48] R. Bellman, Some Problems in the Theory of Dynamic Programming, Econometrica: Journal of the Econometric Society 22 (1954) 37–48.

[49] A. Vakanski, I. Mantegh, A. Irish, F. Janabi-Sharifi, Trajectory Learning for Robot Programming by Demonstration Using Hidden Markov Model and Dynamic Time Warping, IEEE transactions on systems, man, and cybernetics. Part B, Cybernetics : a publication of the IEEE Systems, Man, and Cybernetics Society 42 (2012) 1039–1052.

[50] Y. Wu, Y. Demiris, Hierarchical Learning Approach for One-shot Action Imitation in Humanoid Robots, in: Proceedings of the 11th International Conference on Control Automation Robotics and Vision (ICARCV), IEEE, Singapore, 2010, pp. 453–458.

[51] G. Metta, G. Sandini, D. Vernon, L. Natale, F. Nori, The iCub Humanoid Robot, in: Proceedings of the 8th Workshop on Performance Metrics for Intelligent Systems - PerMIS '08, ACM Press, New York, New York, USA, 2008, p. 50.

[52] G. Metta, L. Natale, F. Nori, G. Sandini, The iCub Project: An Open Source Platform for Research in Embodied Cognition, in: Advanced Robotics and its Social Impacts, IEEE, 2011, pp. 24–26.

[53] T. K. Vintsyuk, Speech Discrimination by Dynamic Programming, Cybernetics 4 (1968) 52–57.

[54] S. P. Chatzis, Y. Demiris, Echo State Gaussian Process, IEEE Transactions on Neural Networks 22 (2011) 1435–45.

[55] S. P. Chatzis, Y. Demiris, The Copula Echo State Network, Pattern Recognition 45 (2012) 570–577.

[56] Y. Wu, Y. Demiris, Efficient Template-based Path Imitation by Invariant Feature Mapping, in: Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics (ROBIO), IEEE, Guilin, China, 2009, pp. 913–918.

[57] A. Shukla, A. Billard, Coupled Dynamical System Based Armhand Grasping Model for Learning Fast Adaptation Strategies, Robotics and Autonomous Systems 60 (2012) 424–440.

[58] Y. Su, Y. Wu, K. Lee, Z. Du, Y. Demiris, Robust Grasping Mechanism for an Under-actuated Anthropomorphic Hand under Object Position Uncertainty, in: Proceedings of the 2012 IEEE-RAS International Conference on Humanoid Robots (Humanoids), IEEE RAS, Osaka, Japan, 2012, pp. 719–725.

[59] Y. Su, Y. Wu, Z. Du, H. Soh, Y. Demiris, Enhanced Kinematic Analysis Model for Dexterous Manipulation with Underactuated Hand, in: Proceedings of the 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, Tokyo, Japan, 2013.