Learning Null Space Projections

Learning Null Space Projections

Hsiu-Chin Lin, Matthew Howard and Sethu Vijayakumar

***Abstract*— Many everyday human skills can be considered in terms of performing some task subject to a set of self-imposed or environmental constraints. In recent years, a number of new tools have become available in the learning and robotics community that allow data from constrained and/or redundant systems to be used to uncover underlying consistent behaviours that may be otherwise masked by the constraints. However, while a wide variety of work for generalisation of movements have been proposed, few have explicitly considered learning the constraints of the motion and ways to cope with unknown environment. In this paper, we propose a method to learn the constraints such that some previously learnt behaviours can be adapted to new environment in an appropriate way. In particular, we consider learning the null space projection matrix of a kinematically constrained system, and see how previously learnt policies can be adapted to novel constraints.**

I. Introduction

Many everyday human skills can be considered in terms of performing some task subject to a set of self-imposed or environmental constraints. For example, when pouring water from a bottle, self-imposed constraints apply to the position and the orientation of the hand so that the water falls within the glass. When wiping a table (Fig. 1), the surface of the table acts as an environmental constraint that restricts the hand movements when in contact with the surface.

A promising way to provide robots with skills is to take examples of human demonstrations and attempt to learn a control policy that somehow capture the behaviours [1], [2], [3]. One common application is to control robot manipulator, for example, given constraint in the end-effector space, produce a set of joint-space movements that can satisfy the constraints [4]. Behaviour may be subject to various constraints that are usually non-linear in actuator space [5], [6]. For example, maintaining balance of the robot (higher priority) while accomplishing an end-effector task (lower priority) [7] or avoiding obstacles [8].

In recent years, a number of new tools have become available in the learning and robotics community that allow data from constrained and/or redundant systems to be used to uncover underlying consistent behaviours that may be otherwise masked by the constraints [9], [10]. While a wide variety of work for generalisation of constrained movements have been proposed, few have explicitly considered learning the *constraints* of the motion and ways to cope with unknown environment.

The ability to deal with unknown environment is favourable since re-learning a behaviour might be time-consuming or require many additional human demonstrations. However, this is not trivial to obtain the precise equations about the environment. For example, adapting the wiping behaviour onto different tables (see Fig. 1) requires prior knowledge of the surface itself. For a car washing robot, the wiping behaviour is acting on various contact constraints, due to the shapes and orientations of the car surface. It would be unfeasible to assume that we have the full knowledge to describe the shape of all areas of different cars. In addition, consider the case of walking on uneven terrain, the feet are constrained by the environment in an unpredictable way.

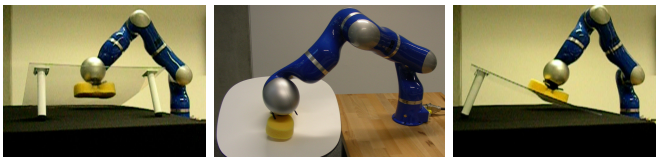


Fig. 1: Examples of wiping on different tables [13]. The behaviour (wiping) is subject to various constraint imposed by the environment where the behaviour is performed (surfaces).

There are a few work related to the estimation of constraint surface, which use force measurements from the end-effector to calculate the plane normal to the constraint surface [11], [12]. However, these are limited to force-control of robot manipulator acting on a smooth surface in a three-dimensional space, and rely on force sensor measurements, which are normally hard to obtain.

In this paper, we propose a method that is able to directly learn the kinematic constraints present in movement observations, as represented by the null space projection matrix of a kinematically constrained system. The proposed approach requires no prior information about either the geometry, or the dimensionality of the constraints, nor does it require information about the control policy underlying observed movement. In addition, we evaluate the proposed approach for learning policies and constraints in problems of varying dimensionality, and illustrate its use in generalisation and predicting behavioural outcomes of policies across different environments.

II. Problem Definition

We consider that the underlying policy is subjected to a set of $\mathcal{S}$-dimensional ($\mathcal{S} \leq \mathcal{Q}$) constraints

$$\mathbf{A}(\mathbf{x}, t)\mathbf{u}(\mathbf{x}, t) = \mathbf{0} \tag{1}$$

where $\mathbf{x} \in \mathbb{R}^{\mathcal{P}}$ represents state, $\mathbf{u} \in \mathbb{R}^{\mathcal{Q}}$ represents the action, and t is time. The *constraint matrix* $\mathbf{A}(\mathbf{x}, t) \in \mathbb{R}^{\mathcal{S} \times \mathcal{Q}}$ is a matrix describing the constraints, which projects the

H. Lin(H.Lin-21@sms.ed.ac.uk) and S. Vijayakumar are at the School of Informatics, Edinburgh University, UK. M. Howard (matthew.j.howard@kcl.ac.uk) is at the Dept. of Informatics, Kings College London, UK.

task-space policy onto the relevant part of the control space. Inverting (1), results in the relation

$$\mathbf{u}(\mathbf{x},t) = \mathbf{N}(\mathbf{x},t)\boldsymbol{\pi}(\mathbf{x}) \tag{2}$$

where $\mathbf{A}^\dagger$ is the Moore-Penrose pseudo-inverse of $\mathbf{A}$,

$$\mathbf{N}(\mathbf{x},t) := (\mathbf{I} - \mathbf{A}(\mathbf{x},t)^\dagger \mathbf{A}(\mathbf{x},t)) \in \mathbb{R}^{\mathcal{Q}\times\mathcal{Q}} \tag{3}$$

is the projection matrix, and $\mathbf{I} \in \mathbb{R}^{\mathcal{Q}\times\mathcal{Q}}$ is the identity matrix. The projection matrix $\mathbf{N}$ projects the null space policy onto the null space of $\mathbf{A}$, which in general, has non-linear dependence on both time and state.

It would be useful to know the decomposition of $\mathbf{A}$, $\mathbf{N}$, and $\boldsymbol{\pi}$; however, the true quantities of those variables are unavailable by assumption. Several studies have been devoted to learning the null space policy $\boldsymbol{\pi}$, but, to the authors' knowledge, none have been able to explicitly estimate $\mathbf{A}$ or $\mathbf{N}$. In [10], for instance, the *null space policy learning* method decomposes the observations $\mathbf{u}$ into two orthogonal components $\mathbf{u} \equiv \mathbf{u}^{\text{ts}} + \mathbf{u}^{\text{ns}}$ such that $\mathbf{u}^{\text{ts}} \equiv \mathbf{A}^\dagger \mathbf{b}$, $\mathbf{u}^{\text{ns}} \equiv \mathbf{N}\boldsymbol{\pi}$, and $\mathbf{u}^{\text{ts}} \perp \mathbf{u}^{\text{ns}}$. This method has been adapted in [14] to learn the decomposition of $\mathbf{u}^{\text{ns}}$, and then estimates the projection matrix $\mathbf{N}$ for analysing human walking. In [15] a method for learning $\mathbf{N}$ is outlined for the special case of a one-dimensional constraint in a 2-DOF system (i.e., where $\mathbf{A} \in \mathbb{R}^{1\times 2}$). In this paper, this work is extended to estimate the projection matrix in a more generic way. Namely, a method is proposed whereby $\mathbf{N}$ can be learnt for $\mathbf{A} \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ where $1 < \mathcal{S} < \mathcal{Q}$.

III. Method

The proposed method works on data given as $\mathcal{N}$ pairs of observed states $\mathbf{x}_n$ and observed actions $\mathbf{u}_n$. It is assumed that (i) the observations can be decomposed as $\mathbf{u} = \mathbf{N}\boldsymbol{\pi}$ (ii) $\mathbf{u}$ are generated using the same null space policy $\boldsymbol{\pi}$, (iii) each observation might have been constrained for some $\mathbf{A} \neq 0$, and (iv) $\mathbf{A}$ (and $\mathbf{N}$) are not explicitly known for any given observation.

The key to the proposed approach is to use properties of the projection matrix $\mathbf{N}$ in order to find $\mathbf{A}$. By definition $\mathbf{u} = \mathbf{N}\boldsymbol{\pi}$, so $\mathbf{u}$ is the vector $\boldsymbol{\pi}$ projected onto the image space of $\mathbf{N}$. However, it is also the case that *the projection of* $\mathbf{u}$ *also lies in this image space*, i.e.,

$$\mathbf{N}\mathbf{u} = \mathbf{u}. \tag{4}$$

Based on this insight, $\mathbf{N}$ can be approximated by seeking an estimate *such that this condition holds*.

Specifically, given samples $\{\mathbf{x}_n, \mathbf{u}_n\}_{n=1}^{\mathcal{N}}$ it is proposed to form an estimate $\tilde{\mathbf{N}}$ that minimises the difference between $\tilde{\mathbf{N}}\mathbf{u}$ and the raw observations $\mathbf{u}$, i.e.,

$$E[\tilde{\mathbf{N}}] = \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\mathbf{u}_n||^2. \tag{5}$$

Using (3), the nth term of (5) can be written

$$\begin{aligned} ||\mathbf{u}_n - (\mathbf{I} - \tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}})\mathbf{u}_n||^2 &= ||\mathbf{u}_n - \mathbf{u}_n + \tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2 \\ &= ||\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2 \end{aligned} \tag{6}$$

where $\tilde{\mathbf{A}} \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ is an estimate of the constraint matrix $\mathbf{A}$.

Expanding the norm $||\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n||^2 = \mathbf{u}_n^\top(\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}})^\top\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n$, and using the identities $(\mathbf{A}^\dagger\mathbf{A})^\top = \mathbf{A}^\dagger\mathbf{A}$ and $\mathbf{A}^\dagger\mathbf{A}\mathbf{A}^\dagger = \mathbf{A}^\dagger$, (5) can be expressed in simplified form

$$E[\tilde{\mathbf{N}}] = \sum_{n=1}^{\mathcal{N}} \mathbf{u}_n^\top\tilde{\mathbf{A}}^\dagger\tilde{\mathbf{A}}\mathbf{u}_n. \tag{7}$$

The constraint projection matrix can therefore be estimated by seeking an estimate $\tilde{\mathbf{A}}$ that minimises (7). Note that, through use of the latter, *no prior knowledge of the underlying* $\boldsymbol{\pi}$*, nor of the true projection matrix* $\mathbf{N}$ *is required.*

A. Representation of $\tilde{\mathbf{A}}$

Until this point, no specific assumptions have been made on the form of the estimated matrix $\tilde{\mathbf{A}}$. Here, an appropriate structure of this matrix is outlined, that ensures that the estimate is interpretable in terms of the constraints. Specifically, $\tilde{\mathbf{A}}$ is assumed to be formed from a set of $\mathcal{S}$ vectors

$$\tilde{\mathbf{A}} = \begin{pmatrix}\boldsymbol{\alpha}_1^\top & \boldsymbol{\alpha}_2^\top & \cdots & \boldsymbol{\alpha}_{\mathcal{S}}^\top\end{pmatrix}^\top \tag{8}$$

where $\boldsymbol{\alpha}_s = (\alpha_{s,1}, \alpha_{s,2}, ..., \alpha_{s,\mathcal{Q}})$ corresponds to the sth constraint in the observations, and $\boldsymbol{\alpha}_i \perp \boldsymbol{\alpha}_j$ for all $i \neq j$.

Since $\tilde{\mathbf{A}}$ consists of linearly independent row vectors, $(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^\top)$ is invertible, and the pseudoinverse has the explicit formula $\tilde{\mathbf{A}}^\dagger = \tilde{\mathbf{A}}^\top(\tilde{\mathbf{A}}\tilde{\mathbf{A}}^\top)^{-1}$, i.e.,

$$\tilde{\mathbf{A}}^\dagger = \begin{pmatrix}\boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \\ \vdots \\ \boldsymbol{\alpha}_{\mathcal{S}}\end{pmatrix}^\top \begin{pmatrix}\boldsymbol{\alpha}_1\boldsymbol{\alpha}_1^\top & \boldsymbol{\alpha}_1\boldsymbol{\alpha}_2^\top & \dots & \boldsymbol{\alpha}_1\boldsymbol{\alpha}_{\mathcal{S}}^\top \\ \boldsymbol{\alpha}_2\boldsymbol{\alpha}_1^\top & \boldsymbol{\alpha}_2\boldsymbol{\alpha}_2^\top & \dots & \boldsymbol{\alpha}_2\boldsymbol{\alpha}_{\mathcal{S}}^\top \\ \vdots & & \ddots & \vdots \\ \boldsymbol{\alpha}_{\mathcal{S}}\boldsymbol{\alpha}_1^\top & \boldsymbol{\alpha}_{\mathcal{S}}\boldsymbol{\alpha}_2^\top & \dots & \boldsymbol{\alpha}_{\mathcal{S}}\boldsymbol{\alpha}_{\mathcal{S}}^\top\end{pmatrix}^{-1}. \tag{9}$$

Noting that $\boldsymbol{\alpha}_i\boldsymbol{\alpha}_i^\top = ||\boldsymbol{\alpha}_i||^2$ and $\boldsymbol{\alpha}_i\boldsymbol{\alpha}_j^\top = 0$ for $i \neq j$, it can be shown that

$$\tilde{\mathbf{A}}^\dagger = \begin{pmatrix}\frac{\boldsymbol{\alpha}_1^\top}{||\boldsymbol{\alpha}_1||^2} & \frac{\boldsymbol{\alpha}_2^\top}{||\boldsymbol{\alpha}_2||^2} & \cdots & \frac{\boldsymbol{\alpha}_{\mathcal{S}}^\top}{||\boldsymbol{\alpha}_{\mathcal{S}}||^2}\end{pmatrix}. \tag{10}$$

From (3), the projection matrix is

$$\tilde{\mathbf{N}} = \mathbf{I} - \begin{pmatrix}\frac{\boldsymbol{\alpha}_1^\top}{||\boldsymbol{\alpha}_1||^2} & \frac{\boldsymbol{\alpha}_2^\top}{||\boldsymbol{\alpha}_2||^2} & \cdots & \frac{\boldsymbol{\alpha}_{\mathcal{S}}^\top}{||\boldsymbol{\alpha}_{\mathcal{S}}||^2}\end{pmatrix}\begin{pmatrix}\boldsymbol{\alpha}_1 \\ \boldsymbol{\alpha}_2 \\ \vdots \\ \boldsymbol{\alpha}_{\mathcal{S}}\end{pmatrix} = \mathbf{I} - \sum_{s=1}^{\mathcal{S}} \hat{\boldsymbol{\alpha}}_s^\top\hat{\boldsymbol{\alpha}}_s \tag{11}$$

where $\boldsymbol{\alpha}_i = ||\boldsymbol{\alpha}_i||\hat{\boldsymbol{\alpha}}_i$ is used. The last equality of (11), makes it clear that the projection matrix is independent of the *magnitude* of each of the $\boldsymbol{\alpha}_s$ since only unit vectors appear in the expression. Hence, only the *direction* of these vectors need be approximated.

B. Forming the Estimate $\tilde{\mathbf{N}}$

In the following, methods for forming the estimate $\tilde{\mathbf{N}}$ are outlined, according to the problem setting.

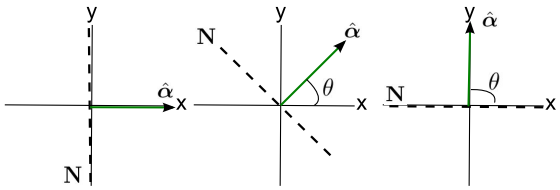


Fig. 2: Examples of (a) $\theta = 0^\circ$, (b) $\theta = 45^\circ$, and (c) $\theta = 90^\circ$ in polar representation and the corresponding null-space (dashed line).

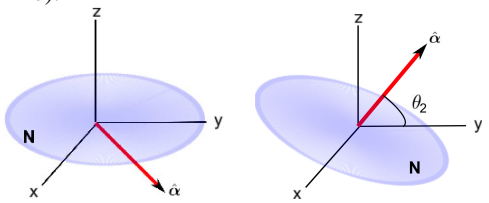


Fig. 3: Examples of $\boldsymbol{\alpha}$ in spherical representation and its null space where $\theta_1 = 45^\circ$, (a) $\theta_2 = 0^\circ$ and (b) $\theta_2 = 45^\circ$.

1) Estimation for $\mathbf{A} \in \mathbb{R}^{1\times 2}$ *:* To begin, consider the simplest case where $\mathbf{A}$ represents a one-dimensional constraint in a two degree of freedom system, i.e., $\mathbf{A} \in \mathbb{R}^{1\times 2}$. As noted in [15], in this case $\hat{\boldsymbol{\alpha}}$ is simply the unit vector

$$\hat{\boldsymbol{\alpha}} = (\cos\theta, \sin\theta) \tag{12}$$

where θ is the angle encoding the direction (see Fig. 2).

A simple method for forming the estimate $\tilde{\mathbf{A}}$ is to perform a line search whereby a set of $\mathcal{L}$ sample θ are generated, and that which minimises the error criterion (7) is chosen for the approximation. Note that, the search need only be restricted to the half-space $0 \leq \theta < \pi$ since this covers all possible cases (the projections corresponding to the half-space $\pi \leq \theta < 2\pi$ are identical to those in the former).

2) Estimation for $\mathbf{A} \in \mathbb{R}^{1\times \mathcal{Q}}$ *with* $\mathcal{Q} > 2$*:* For problems with $\mathcal{Q} > 2$ but with $\mathcal{S} = 1$ the same approach can be applied by extending the $\hat{\boldsymbol{\alpha}}$ in (12) to handle higher degrees of freedom (DOF).

For example, to extend the formulation to $\tilde{\mathbf{A}} \in \mathbb{R}^{1\times 3}$, the search should be conducted over the set of possible unit vectors $\hat{\boldsymbol{\alpha}} \in \mathbb{R}^3$. Using spherical coordinates, the latter can be represented as

$$\hat{\boldsymbol{\alpha}} = (\cos\theta_1, \sin\theta_1\cos\theta_2, \sin\theta_1\sin\theta_2) \tag{13}$$

where $\theta_1 \in (0, \pi]$ and $\theta_2 \in (0, \pi]$ encode the direction of the vector in the three dimensional space. For instance, in Fig. 3, the additional variable θ_2 specifies the angle between $\boldsymbol{\alpha}$ and xz-plane.

Similarly, for $\mathcal{S} = 4$ the vector $\hat{\boldsymbol{\alpha}}$ can be represented by three angles θ_1, θ_2 and θ_3 with elements as indicated in the third column of Table I.

More generally, for $\mathcal{Q} > 2$, the unit vector $\hat{\boldsymbol{\alpha}} = (\hat{\alpha}_1, \hat{\alpha}_2, \cdots, \hat{\alpha}_\mathcal{Q})$ can be represented by $\mathcal{Q}-1$ parameters

	$\mathbb{R}^2$	$\mathbb{R}^3$	$\mathbb{R}^4$
$\hat{\alpha}_1$	$\cos\theta_1$	$\cos\theta_1$	$\cos\theta_1$
$\hat{\alpha}_2$	$\sin\theta_1$	$\sin\theta_1\cos\theta_2$	$\sin\theta_1\cos\theta_2$
$\hat{\alpha}_3$	-	$\sin\theta_1\sin\theta_2$	$\sin\theta_1\sin\theta_2\cos\theta_3$
$\hat{\alpha}_4$	-	-	$\sin\theta_1\sin\theta_2\sin\theta_3$

TABLE I: Representation of unit vector in $\mathbb{R}^2$, $\mathbb{R}^3$, and $\mathbb{R}^4$.

$\boldsymbol{\theta} = (\theta_1, \theta_2, \cdots, \theta_{\mathcal{Q}-1})^\top$ where

$$\begin{aligned}
\hat{\alpha}_1 &= \cos\theta_1 \\
\hat{\alpha}_2 &= \sin\theta_1\cos\theta_2 \\
\hat{\alpha}_3 &= \sin\theta_1\sin\theta_2\cos\theta_3 \\
&\vdots \\
\hat{\alpha}_{\mathcal{Q}-1} &= \prod_{q=1}^{\mathcal{Q}-2}\sin\theta_q\cos\theta_{\mathcal{Q}-1} \\
\hat{\alpha}_\mathcal{Q} &= \prod_{q=1}^{\mathcal{Q}-1}\sin\theta_q
\end{aligned} \tag{14}$$

Note that, in this case, if an exhaustive search over an $\mathcal{L}$-sided grid in the $(\mathcal{Q}-1)$-dimensional space of parameters is used for selecting $\boldsymbol{\theta}$, the sequential running time increases to $\mathcal{O}(\mathcal{L}^{\mathcal{Q}-1})$.

3) Estimation for $\mathbf{A} \in \mathbb{R}^{\mathcal{S}\times\mathcal{Q}}$ *with* $\mathcal{Q} > 2$ *and* $\mathcal{S} > 1$*:* For systems subject to multidimensional constraints, $\mathbf{A}$ consists of a set of $\mathcal{S}$ orthogonal vectors as in (8), see §III-A. The same optimisation approach can be used to form the estimate $\tilde{\mathbf{N}}$ as in the preceding sections, namely, by defining an orthogonal unit vector $\hat{\boldsymbol{\alpha}}_s$ for each of the $\mathcal{S}$ constraints, and seeking the optimal parameters $\boldsymbol{\theta}_s = (\theta_{s,1}, \theta_{s,2}, \cdots, \theta_{s,\mathcal{Q}-1})^\top$ that minimise (7).

In this case, an iterative approach to learning may be employed, whereby a series of of constraint vectors $\hat{\boldsymbol{\alpha}}_s$ are fitted to the data to form an estimate $\tilde{\mathbf{A}}$, where an $(s+1)^{th}$ vector is *only added if it does not reduce the fit under* (7). This exploits a second property of the projection matrix $\mathbf{N}$, namely that for multidimensional constraints the total projection can be decomposed into a set of unidimensional projections

$$\mathbf{N} = \mathbf{N}_1\mathbf{N}_2\cdots\mathbf{N}_\mathcal{S}, \tag{15}$$

or equivalently,

$$\mathbf{u} = \mathbf{N}\boldsymbol{\pi} = \mathbf{N}_1(\mathbf{N}_2\cdots(\cdots\mathbf{N}_\mathcal{S}\boldsymbol{\pi})\cdots)) \tag{16}$$

through (2). This suggests that a reasonable approximation $\tilde{\mathbf{N}}$ may be formed by (i) first finding the optimal $\hat{\boldsymbol{\alpha}}_1$ (i.e., $\boldsymbol{\theta}_1^*$) under (7) through the fitting procedure described in §III-B.2, then (ii) finding the optimal $\hat{\boldsymbol{\alpha}}_2$ (i.e., $\boldsymbol{\theta}_2^*$), subject to $\hat{\boldsymbol{\alpha}}_1 \perp \hat{\boldsymbol{\alpha}}_2^*$, and (iii) repeating until the addition of a new constraint $\hat{\boldsymbol{\alpha}}_{s+1}^*$ fails to reduce the error under (7) any further. The process is summarised in Algorithm 1.

C. Evaluation Criteria

The goal of this work is to predict the projection matrix $\mathbf{N}$ underlying the constrained observations in order that these may be reproduced through a suitable learning scheme (e.g., [9], [16]). For testing the performance of learning, therefore, the following evaluation criteria may be defined.

Algorithm 1 Projection Matrix Learning

Input: State, action samples $\{\mathbf{u}_n\}_{n=1}^{\mathcal{N}}$
Output: $\hat{\boldsymbol{\alpha}}_i$: the set of constraint vectors

```
Estimate α̂_1 by minimising (7). Set s ← 1.
while E[Ñ] in (7) is not increasing do
    s ← s + 1.
    Learn α*_s minimising (7) such that α*_s ⊥ α_i ∀i < s.
    Set Ã ← [α̂_1^⊤, ··· , α̂_s^⊤]^⊤.
end while
Return α̂_i.
```

1) Normalised Projected Policy Error: This error measure measures the difference between the policy subject to the true constraints, and that of the policy subject to the estimated constraints. Formally, the normalised projected policy error (NPPE) can be defined as

$$E_{PPE} = \frac{1}{\mathcal{N}\sigma_{\mathbf{u}}^2} \sum_{n=1}^{\mathcal{N}} ||\mathbf{N}\boldsymbol{\pi}_n - \tilde{\mathbf{N}}\boldsymbol{\pi}_n||^2 \tag{17}$$

where $\mathcal{N}$ is the number of data points, $\boldsymbol{\pi}_n$ are samples of the policy, and $\mathbf{N}$ and $\tilde{\mathbf{N}}$ are the true and the learnt projection matrices, respectively. The error is normalised by the variance of the observations under the true constraints $\sigma_{\mathbf{u}}^2$. Note that, since $\mathbf{u} = \mathbf{N}\boldsymbol{\pi}$, (17) can also be written

$$E_{PPE} = \frac{1}{\mathcal{N}\sigma_{\mathbf{u}}^2} \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\boldsymbol{\pi}_n||^2 \tag{18}$$

which corresponds directly to the *normalised constrained policy error* (NCPE) [9], between the constrained observations and the policy *subject to the learnt constraints*. Note that, (17) can only be computed given the ground truth $\boldsymbol{\pi}_n$ and $\mathbf{N}$, so is used here primarily for validation purposes.

2) Normalised Projected Observation Error: To evaluate the fit in absence of samples of $\boldsymbol{\pi}_n$ and $\mathbf{N}$, an alternative criterion must be used. It is proposed, therefore, to instead use the *normalised projected observation error* (NPOE),

$$E_{POE} = \frac{1}{\mathcal{N}\sigma_{\mathbf{u}}^2} \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}}\mathbf{u}_n||^2. \tag{19}$$

The NPOE indicates the quality of fit of $\tilde{\mathbf{N}}$, i.e., the extent to which the constraints in the training data are captured by the model, reaching zero only[1] when the model exactly satisfies the condition (4).

IV. Evaluation

In this section, some numerical results are presented to evaluate the learning performance.

A. Toy Example

Our first experiment demonstrates our approach for learning null space projections from constrained data. For this, we set up a two-dimensional system with a one-dimensional constraint (i.e., $\mathbf{A} \in \mathbb{R}^{1\times 2}$). As ground truth null space policies $\boldsymbol{\pi}$, we considered three difference policy types:

[1]Assuming $\mathbf{u}_n \neq \mathbf{0}$ for some n.

Policy	NPPE	NPOE
Linear	2.09 ± 0.004	1.04 ± 0.05
Limit-cycle	2.09 ± 0.004	1.05 ± 0.03
Sinusoid	2.11 ± 0.011	1.09 ± 0.02

TABLE II: Normalised PPE and POE in predicting the projection matrix. Results are (mean±s.d.)$\times 10^{-6}$ over 50 trials.

(i) a linear policy: $\boldsymbol{\pi} = -\mathbf{L}(\mathbf{x} - \mathbf{x}^*)$ where $\mathbf{L}$ is a positive definite gain matrix.
(ii) a limit-cycle policy: $\dot{r} = r(\rho - r^2)$ with radius $\rho = 0.1\,m$, angular velocity $\dot{\theta} = -2$ rad/s, where r and θ are the polar representation of the state, i.e., $\mathbf{x} = (r\cos\theta, r\sin\theta)^\top$.
(iii) a sinusoidal policy: $\boldsymbol{\pi} = 0.5(\cos x_1 \cos x_2, \sin x_1 \sin x_2)^\top$

The training data consists of 50 data points, drawn uniform-randomly across the space $(\mathbf{x})_i \sim \mathcal{U}(-2, 2)$, $i \in \{1, 2\}$ and subjected to a 1-D constraint $\mathbf{A} = \hat{\boldsymbol{\alpha}} \in \mathbb{R}^{1\times 2}$, in the direction of the unit vector $\hat{\boldsymbol{\alpha}}$. The latter is drawn uniform-randomly, $\theta \sim \mathcal{U}(0, \pi]\, rad$, at the start of each trial of learning. A sample data set for the limit-cycle policy is presented in Fig. 4 (left) where projected sample points $\mathbf{u}_n$ are shown in red, with the corresponding unconstrained null space policy predictions $\boldsymbol{\pi}$ overlaid in grey.

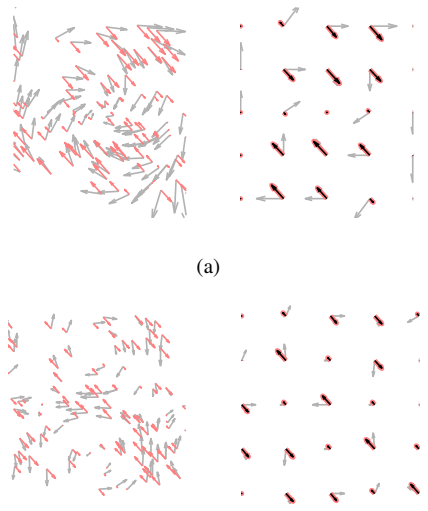
The projection matrix $\tilde{\mathbf{N}}$ is then learnt using this data through minimisation of the objective function (7) according to the scheme outlined in §III. For this, a grid search over 180 different values for θ, uniformly spaced between 0 and π is conducted to recover the optimal θ^*.

The experiment is repeated 50 times and the average NPPE and NPOE are evaluated on a set of 500 test data points, generated through the same procedure as described above.

Table II summarises the normalised PPE (17) and normalised POE (19) for each policy. The results are average using the hold-out testing over 50 trails. We can see that, our method can learn a good approximation of the projection matrix in terms of both PPE and POE, and the performance is not significantly affected by the policy.

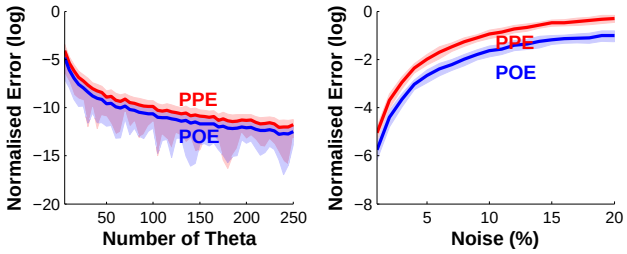
In Fig. 4, the predictions of the policy under the true constraint $\mathbf{N}\boldsymbol{\pi}$ (black), and the learnt projection matrix $\tilde{\mathbf{N}}\boldsymbol{\pi}$ (red) are plotted for the limit cycle policy. As can be seen, there is good agreement between the two, verifying that in using the learnt constraint, there is little degradation in predicting constrained motion.

To further characterise the performance of the proposed approach, we also looked at the effect of varying the density of sample points in the grid search for θ for the limit cycle policy. We test our method for $10 < \mathcal{L} < 250$. The results over 50 trials are plotted in Fig. 5a. It can be seen that the NPPE and NPOE rapidly decrease as the number of θ sampled increases (please note the log scale). This is to be expected, since a higher resolution grid allows the learner to form a more accurate estimate of the constraint direction. Note, however, that even at relatively course sampling ($\mathcal{L} < 50$), the error is still very low.



(a) (b)

Fig. 4: A visualisation of the (a) limit-cycle and (b) sinusoid data. The left figures are the training data and the right figures are the testing data. The colours denote the true policy (grey), the true constrained policy (red), and the predicted constrained policy (black).



(a) (b)

Fig. 5: Normalised PPE and POE for (a) increasing number of sampling θ and (b) increasing noise levels in the observed $\mathbf{u}$. Curves are mean$\pm$s.d. over 50 trials.

We also test how the levels of noise present in the training data affect the performance of our method. For this, we contaminated the limit-cycle policy $\boldsymbol{\pi}$ with Gaussian noise, the scale of which we varied to match up to 20% of the data. The resulting NPPE and NPOE follows the noise level, as plotted in Fig. 5b. It should be noted, however, that the error is still relatively low (NPPE$< 10^{-2}$), even when the noise is as high as 5% of the variance of the data.

B. Kuka Lightweight Robot

The goal of this experiment is to assess the performance of the proposed approach for learning with higher degrees of freedom, and more realistic constraints with varying dimensionality.

For this, constrained motion data from a kinematic simulation of the 7-DOF Kuka Lightweight Robot (LWR-III) is used (Fig. 1). Here, the state and action space correspond to the joint angles and velocities, respectively, i.e., $\mathbf{x}, \mathbf{u} \in \mathbb{R}^7$, and data is gathered from the arm subject to varying constraints on its end-effector motion.

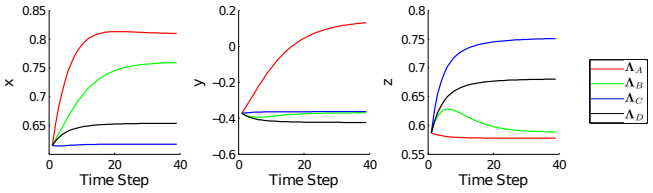


Fig. 6: Example trajectories generated from 4 different types of constraints

Specifically, constraints are imposed on motion in the task space $\mathbf{r} = (r_x, r_y, r_z)^\top$ where r_x, r_y and r_z denote the translational coordinates of the end-effector. Mathematically, they are described through constraint matrices of the form

$$\mathbf{A}(\mathbf{x}) = \boldsymbol{\Lambda}\mathbf{J}(\mathbf{x}) \tag{20}$$

where $\mathbf{J}(\mathbf{x}) \in \mathbb{R}^{3\times 7}$ is the manipulator Jacobian, and $\boldsymbol{\Lambda} \in \mathbb{R}^{\mathcal{S}\times 3}$ is a matrix specifying the coordinates to be constrained. For example, choosing $\boldsymbol{\Lambda} = (0, 0, 1)$, ensures that the end-effector does not move in the vertical (r_z) direction (a 'one-dimensional' constraint). Using $\boldsymbol{\Lambda} = (\boldsymbol{\lambda}_1^\top, \boldsymbol{\lambda}_2^\top)^\top$ with $\boldsymbol{\lambda}_1 = (1, 0, 0)$ and $\boldsymbol{\lambda}_2 = (0, 1, 0)$ prevents movement of the end-effector in both r_x and r_y. Note that, neither the dimensionality of the constraint, nor the subspace in which it acts, is provided a priori to the learner. The experimental procedure is as follows.

Data is gathered by recording motion of the arm, subject to different constraints, from a number of random start states. The latter are drawn randomly from the robot joint space in the half-range of the joint limits, i.e., $(\mathbf{x})_i \sim \mathcal{U}(-0.5(\mathbf{x}^{max})_i, 0.5(\mathbf{x}^{max})_i)$ where $\mathbf{x}^{max} = (170°, 120°, 170°, 120°, 170°, 120°, 170°)^\top$. For each start state, a trajectory is recorded from a point attractor policy $\boldsymbol{\pi} = -\mathbf{L}(\mathbf{x} - \mathbf{x}^*)$ where $\mathbf{x}^* = \mathbf{0}$ and $\mathbf{L} = \mathbf{I}$ under the active constraint. In each data set, the latter consisted of one of the following constraints:

(i) $\boldsymbol{\Lambda}_A = (0, 0, 1)$,
(ii) $\boldsymbol{\Lambda}_B = (0, \sin\frac{\pi}{3}, \cos\frac{\pi}{3})$,
(iii) $\boldsymbol{\Lambda}_C = ((1, 0, 0), (0, 1, 0))^\top$, and
(iv) $\boldsymbol{\Lambda}_D = (\boldsymbol{\lambda}_1, \boldsymbol{\lambda}_2)^\top$ with $\boldsymbol{\lambda}_1 = (0, \sin\frac{\pi}{3}, \cos\frac{\pi}{3})$ and $\boldsymbol{\lambda}_2 = (\sin\frac{\pi}{2}\sin\frac{\pi}{3}, \sin\frac{\pi}{2}\cos\frac{\pi}{3}, \cos\frac{\pi}{2})$.

These correspond to various real world tasks, for example, $\boldsymbol{\Lambda}_A$ applies when interacting with a flat (horizontal) surface (e.g., as when wiping or writing on a table). The $\boldsymbol{\Lambda}_B$ corresponds to the case where the surface is inclined by $\pi/3\,rad$. In this way, data sets containing $\mathcal{K} = 100$ trajectories of length $\mathcal{T} = 40$ sample points are generated, for each of the constraints are collected, and a random $\mathcal{N} = 10$ data points are selected as training data. Examples of end-effector trajectories for each constraint are plotted in Fig. 6.

With this training data, Algorithm 1 is then used to form an estimate of $\boldsymbol{\Lambda}$. For each $\boldsymbol{\theta}_{s,i}$ defining the unit vectors $\hat{\boldsymbol{\alpha}}_s$, 180 different values are tested, uniformly spaced in the range $\boldsymbol{\theta}_{s,i} \in [0, \pi)$. Note that, as described in §III, constraints $\hat{\boldsymbol{\alpha}}_s$ are iteratively added, subject to $\alpha_s^* \perp \alpha_i \, \forall i < s$ until an increase in (7) is seen. In this way, no prior knowledge of the constraint dimensionality is needed for learning.

	NUPE	NCPE	NMSE
$\mathbf{\Lambda}_A$	7.50 ± 0.14	6.00 ± 0.09	6.00 ± 0.09
$\mathbf{\Lambda}_B$	7.26 ± 0.12	5.77 ± 0.10	5.77 ± 0.10
$\mathbf{\Lambda}_C$	6.74 ± 0.07	4.45 ± 0.05	4.45 ± 0.05
$\mathbf{\Lambda}_D$	6.72 ± 0.09	4.40 ± 0.08	4.40 ± 0.08

TABLE III: Normalised UPE and CPE for generalising the joint-limit avoidance policy and the NMSE when applied the learnt constraints (from linear attractor policy) on the learnt policies. The results are (mean$\pm$s.d.)$\times 10^{-1}$ over 50 trails with different data sets.

To assess the performance, the errors are computed according to the evaluation criteria described in §III-C on independent test data generated according to the same procedure as above.

The experiment was repeated for 50 trials for each data set, and each constraint. In all cases, the proposed method was able to estimate the constraint with NPPE $< 10^{-3}$ and NPOE $< 10^{-4}$. To verify the effectiveness of the stopping condition, we also evaluated the error under (7) for each new $\hat{\boldsymbol{\alpha}}_s$ added. For example, for $\mathbf{\Lambda}_C$, the correct constraint dimensionality is $\mathcal{S} = 2$, and the average errors at the first two iterations of learning this constraint are lower than 10^{-10}. However at $s = 3$ the average error jumps to 0.281 ± 0.218, signifying that the dimensionality of the learnt constraint is higher than the ground truth, and there is no need to search further.

In many scenarios, it would be useful to predict the behavioural outcomes of using a new policy $\boldsymbol{\pi}'$ to a previously seen environment where the constraints $\tilde{\mathbf{N}}$ have been learnt. This is especially the case for evaluating the use of previously untested policies (e.g., those resulting from optimisation or reinforcement learning) may be too risky to directly evaluate on the robot prior to simulation.

To test the use of the proposed approach for this, we also evaluated the quality of the learnt constraint, in predicting the constrained actions of a new policy, not present in the training data. In the results reported here, the accuracy in predicting the constrained action of a joint-limit avoidance policy, $\boldsymbol{\pi}(\mathbf{x}) = -0.5 \bigtriangledown \Phi(\mathbf{x})$ with the potential given by $\Phi(\mathbf{x}) = \sum_{i=1}^{7} |x_i|^2$, under the learnt constraint, is evaluated. We apply the constraints learnt from the linear attractor policy to this policy, and the resulting NPPE and NPOE were $< 10^{-3}$, indicating good prediction performance of the behavioural outcomes of the new policy under the learnt constraints.

C. Combined Constraint and Policy Learning

The goal of this final experiment is to demonstrate the use of the proposed approach in the context of constrained motion imitation learning. As described in §I, in many every day behaviours, it is useful to be able to form an estimate both of the policy underlying motion, as well as the constraint itself. In this way, generalisation can be achieved both across constraints (i.e., applying the learnt policy to new constraints), as well as within constraints (i.e., applying new policies to the learnt constraint). To test this, the proposed approach is combined with that proposed in [9] for learning models of both $\boldsymbol{\pi}$ and $\mathbf{N}$.

Using the data collected under constraints $\mathbf{\Lambda}_A$, $\mathbf{\Lambda}_B$, $\mathbf{\Lambda}_C$ and $\mathbf{\Lambda}_D$ from the preceding experiment (ref. §IV-B), constraint consistent learning [9] is used to estimate a model of the policy $\boldsymbol{\pi}$. In more detail, we used linear regression to learn a policy $\tilde{\boldsymbol{\pi}}$ for the data set generated from the linear policy. For the joint limit avoidance policy, we used parametric models consisting of 250 Gaussian RBFs with centres chosen according to k-means and with widths taken as the mean of the distances between centres. To assess learning performance, the normalised unconstrained policy error (NUPE), constrained policy error (NCPE) and mean squared error (NMSE) are evaluated (see Appendix, equations (21), (22) and (23), respectively) over 50 trials of learning.

For the linear attractor policy, we were able to obtain nearly perfect generalisation with the NUPE, NCPE and NMSE all $< 10^{-9}$. The result confirms that with good enough estimation of the constraints and policies, we can accurately predict the constrained policy, even without prior knowledge of the true policy $\boldsymbol{\pi}$. The joint-limit-avoidance policy turned out to be a harder problem to learn with relatively high NMSE (ref. Table III).

We also evaluated whether we can use these learnt behaviours in a new environment which was not present in the data for training the policies. For instance, given demonstrated wiping motion on surface with 0° and 30° inclination (e.g., $\mathbf{\Lambda}_A$ and $\mathbf{\Lambda}_B$), can we adapt the learnt behaviour to another surface with a different slope? For this, we created a new constraint $\mathbf{\Lambda}_E = (\sin\frac{\pi}{3}\sin\frac{\pi}{4}, \cos\frac{\pi}{3}\sin\frac{\pi}{4}, \cos\frac{\pi}{4})$ and followed the same procedure to generate trajectories. We then approximated the constraint, and applied the learnt policies to predict the behavioural outcomes.

Fig. 7 shows the end-effector position when using the learnt $\mathbf{\Lambda}_E$ with the linear policy (Fig. 7a) and the joint-limit avoidance policy (Fig. 7b). The plots from the left to the right are the visualisation in x, y, and z position of the end-effector. The red colour denotes the true constrained movement generated from the true policy ($\mathbf{N}\,\boldsymbol{\pi}$), the black colour denotes the estimated movement when the learnt constraint is applied to the true policy ($\tilde{\mathbf{N}}\,\boldsymbol{\pi}$), and the blue colour denotes the estimation by learning both constraint and the policy ($\tilde{\mathbf{N}}\,\tilde{\boldsymbol{\pi}}$).

For the joint-limit avoidance policy (Fig. 7b), there is a deviation in the end-effector position when applying the learnt constraint on the learnt policy (the blue trajectories), due to the errors in learnt policy. On the other hand, accurate estimation of both constraints and policy, as is the case for the linear policy (Fig. 7a), enables a close reproduction of the ground truth end-effector movements in all directions.

V. Conclusion

In this paper, we explore the problem of learning the constraints from movement observations in uncertain environments, in order that the outcomes of behaviours can be predicted and adapted to new environments in an appropriate way. In particular, we consider how the null space projection matrix of a kinematically constrained system, and have developed a method by which that matrix can be approximated in

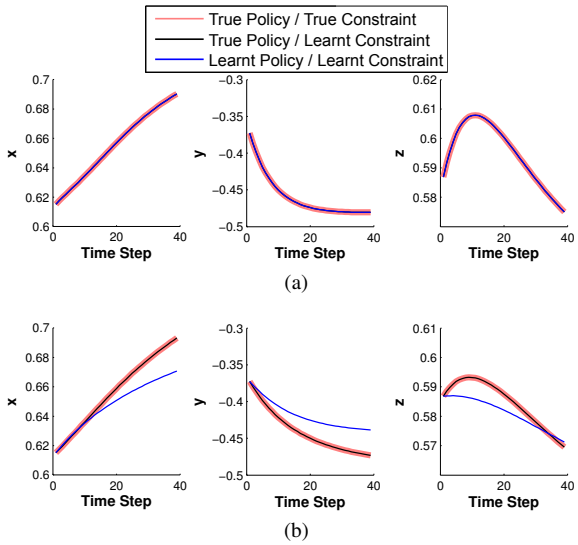


Fig. 7: Visualisation of end-effector trajectories from (a) linear policy and (b) joint-limit avoidance policy. From the left to the right are the x, y, and z position of the end-effector. The colours denote constrained movement generated by the true constraint and the true policy (red), the constrained movement by applying the learnt constraint on the true policy (black), and by applying the learnt constraint on the learnt policy (blue).

the absence of any prior knowledge either on the underlying movement policy, or the geometry or dimensionality of the constraints.

Our evaluations have demonstrated the effectiveness of the proposed approach on problems of differing dimensionality, and with different degrees of non-linearity. The have also validated the use of our method in the generalisation across constraints (i.e., applying the learnt policy to new constraints), as well as within constraints (i.e., applying new policies to the learnt constraint).

For future research, we plan to use the proposed method on human data where the true policy and constraint are both unknown, and to study variants of the approach that may improve its efficiency through iterative learning approaches. With some modification, we also aim to apply the proposed method to analyse the constraints inherent in human walking.

APPENDIX

Unconstrained policy error : Our primary criteria to evaluate a learnt policy is the *normalised unconstrained policy error (UPE)*, which directly compares the true and the learnt policy

$$E_{UPE} = \frac{1}{N\sigma_{\boldsymbol{\pi}}^2} \sum_{n}^{N} || \boldsymbol{\pi}_n - \tilde{\boldsymbol{\pi}}_n ||^2 \qquad (21)$$

where $\sigma_{\boldsymbol{\pi}}$ is the standard deviation of the true policy.

Constrained Policy Error: In some cases, it may be that the variation in constraints is insufficient to fully uncover the true policy $\boldsymbol{\pi}$. However, in such cases, where the constraints exhibit little variation, there may be no need to uncover the hidden components of the fully unconstrained policy (since those components are anyway eliminated by the constraints in normal circumstances). In such circumstances, an alternative quality measure is the *normalised constrained policy error (CPE)*

$$E_{CPE} = \frac{1}{N\sigma_{\boldsymbol{\pi}}^2} \sum_{n}^{N} || \mathbf{u}_n^{ns} - \mathbf{N}_n \tilde{\boldsymbol{\pi}}_n ||^2 \qquad (22)$$

that measures the difference between the data and the estimated policy, when the latter is projected by the same constraints as in the training data.

Normalised Mean-Squared Error: In many everyday behaviours, we will not have access to the true projections nor the true policy. An alternative is to evaluate how well our learnt projection can reproduce the demonstrated motion without any prior knowledge. Assuming that the policy has been estimated in some way, we measure the distance between the observations and the estimated policy subject to the estimated constraints, namely,

$$E_{MSE} = \frac{1}{\mathcal{N}\sigma_{\boldsymbol{\pi}}^2} \sum_{n=1}^{\mathcal{N}} ||\mathbf{u}_n - \tilde{\mathbf{N}} \tilde{\boldsymbol{\pi}} ||^2. \qquad (23)$$

REFERENCES

[1] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, *Robot programming by demonstration*. MIT Press, 2007.

[2] J. Craig, P. Hsu, and S. Sastry, "Adaptive control of mechanical manipulators," *Int. J. Robotics Research*, vol. 6, no. 2, pp. 16–28, 1987.

[3] M. Kawato, "Feedback-error-learning neural network for supervised motor learning," in *Advanced Neural Computers*, R. Eckmiller, Ed. Elsevier, 1990, pp. 365–372.

[4] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE J. Robotics & Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[5] M. Gienger, H. Janssen, and C. Goerick, "Task-oriented whole body motion for humanoid robots," *2005 5th IEEE-RAS Int. Conf. Humanoid Robots*, 2005.

[6] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *Int. J. Humanoid Robotics*, 2004.

[7] L. Sentis and O. Khatib, "Synthesis of whole-body behaviors through hierarchical control of behavioral primitives," *Int. J. Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[8] M. Stilman and J. Kuffner, "Planning among movable obstacles with artificial constraints," *Int. J. Robotics Research*, vol. 27, no. 11-12, pp. 1295–1307, 2008.

[9] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "A novel method for learning policies from variable constraint data," *Autonomous Robots*, 2009.

[10] C. Towell, M. Howard, and S. Vijayakumar, "Learning nullspace policies," in *IEEE Int. Conf. Intelligent Robots & Systems*, 2010.

[11] M. Blauer and P. Belanger, "State and parameter estimation for robotic manipulators using force measurements," *IEEE Transactions on Automatic Control*, vol. 32, no. 12, 1987.

[12] T. Yoshikawa, "Dynamic hybrid position/force control of robot manipulators," *IEEE J. Robotics & Automation*, vol. 3, no. 5, 1987.

[13] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "Learning potential-based policies from constrained motion," in *IEEE Int. Conf. Humanoid Robots*, 2008.

[14] H. Lin, M. Howard, and S. Vijayakumar, "A novel approach for representing and generalising periodic gaits," *Robotica*, vol. 32, pp. 1225–1244, 2014.

[15] H. Lin, M. Howard, and S. Vijayakumar, "Generalising walking gaits across subjects and walking speeds," in *IEEE Int. Conf. Biomedical Robotics & Biomechatronics*, 2014.

[16] M. Howard, S. Klanke, M. Gienger, C. Goerick, and S. Vijayakumar, "Robust constraint-consistent learning," in *IEEE Int. Conf. Intelligent Robots & Systems*, 2009.