Applied Bionics and Biomechanics
Vol. 00, No. 00, Month 200x, 1–24

RESEARCH ARTICLE

Behaviour Generation in Humanoids by Learning Potential-based Policies from Constrained Motion

Matthew Howard, Stefan Klanke,
Michael Gienger, Christian Goerick and Sethu Vijayakumar
(*Received 00 Month 200x; final version received 00 Month 200x*)

Movement generation that is consistent with observed or demonstrated behaviour is an efficient way to seed movement planning in complex, high dimensional movement systems like humanoid robots. We present a method for learning potential-based policies from constrained motion data. In contrast to previous approaches to direct policy learning, our method can combine observations from a variety of contexts where different constraints are in force, to learn the underlying unconstrained policy in form of its potential function. This allows us to generalise and predict behaviour where novel constraints apply. We demonstrate our approach on systems of varying complexity, including kinematic data from the ASIMO humanoid robot with 22 degrees of freedom.

1. Introduction

A wide variety of everyday human skills can be framed in terms of performing some task subject to constraints imposed by the physical environment (Ohta et al 2004; Svinin et al 2005). Examples include opening a door, pulling out a drawer or stirring soup in a saucepan.

In a more generic setting, constraints may take a much wider variety of forms. For example, in climbing a ladder, the constraint may be on the centre of mass or the tilt of the torso of the climber to prevent over-balancing. Alternatively, in problems that involve control of contacts such as manipulation or grasping of a solid object, the motion of fingers is constrained during the grasp by the presence of the object (Sapio et al 2006; Park and Khatib 2006). Also in systems designed to be highly competent and adaptive, such as humanoid robots (Fig. 1), behaviour may be subject to a wide variety of constraints (Sentis and Khatib 2006, 2005; Gienger et al 2005; Sapio et al 2005; Sentis and Khatib 2004), usually non-linear in actuator space and often discontinuous. Consider the task of running or walking on uneven terrain: the cyclic movement of the legs of the runner is constrained by the impact of the feet on the ground in a dynamic, discontinuous and unpredictable way. A promising approach to providing robots with such skills as running and opening doors is to take examples of motion from existing demonstrators (e.g., from humans) and attempt to learn a control policy that somehow captures the desired behaviour (Calinon and Billard 2007; Billard et al 2007; Alissandrakis et al 2007; Grimes et al 2007; Chalodhorn et al 2006; Grimes et al 2006; Takano et al 2006; Schaal et al 2003; Inamura et al 2004; Ijspeert et al 2003). An important

M. Howard, S. Klanke and S. Vijayakumar are with the School of Informatics, University of Edinburgh, Edinburgh EH9 3JZ, United Kingdom. E-mail: matthew.howard@ed.ac.uk.
M. Gienger and C. Goerick are with the Honda Research Institute Europe GmbH, Offenbach/Main D-63073, Germany.
Manuscript received XXXXX XX, XXXX; revised XXXXXXX XX, XXXX.

ISSN: 1741-5977 print/ISSN 1741-5985 online

DOI: 10.1080/1741597YYxxxxxxxx
http://www.informaworld.com

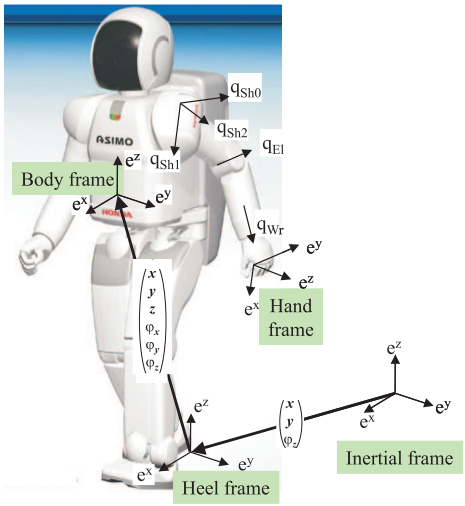


Figure 1. Kinematic model of the ASIMO humanoid robot (Gienger et al 2005). In our experiments 22 upper body degrees of freedom were used (2×7 DOF arms, 2 DOF head, 6 DOF torso), with the heel frame fixed.

component of this is the ability to deal with the effect of constraints and the apparent variability in the observed movement induced by these constraints. For example, one wishes to learn a policy that allows one not only to open a specific door of a particular size (e.g. constraining the hand to a curve of a particular radius), but rather to open many doors of varying sizes (or radii).

The focus in this paper is on modelling control policies subject to a specific class of constraints on motion, with the aim of finding policies that can generalise *over different constraints*. We take a direct policy learning (DPL) approach (Guenter et al 2007; Chalodhorn et al 2006; Nakanishi et al 2004; Schaal et al 2003; Atkeson and Schaal 1997; Mussa-Ivaldi 1997) whereby we attempt to learn a continuous model of the policy from motion data. While DPL has been studied for a variety of control problems in recent years[1], crucially these problems involved policies that are either directly observable from motion data, i.e. unconstrained policies, or policies subject to identical constraints in every observation, in which case the constraints can be absorbed into the policy itself. The difference here is that we consider observations from policies projected into the nullspace of a set of dynamic, non-linear constraints, and that these constraints may *change between observations*, or even during the course of a single observation.

Our strategy for this is to attempt to consolidate movement observations under different specific constraints to find the underlying *unconstrained policy* common to all. Learning the latter enables generalisation since we can apply new constraints to predict behaviour in novel scenarios. In general, however, learning (unconstrained) policies from constrained motion data is a formidable task. This is due to (i) the *non-convexity* of observations under different constraints, and; (ii) *degeneracy* in the set of possible policies that could have produced the movement under the constraint (Howard et al 2008; Howard and Vijayakumar 2007). However, despite these hard analytical limits, we will show that it is still possible to find a good approximation of the unconstrained policy given observations under the right conditions. We take advantage of recent work in local dimensionality reduction (Verbeek et al 2004) to propose a method that (i) given observations under a sufficiently rich set

[1]For a review on DPL , please see (Billard et al 2007) and references therein.

of constraints reconstructs the fully unconstrained policy; (ii) given observations under an impoverished set of constraints learns a policy that generalises well to constraints of a similar class, and; (iii) given 'pathological' constraints will learn a policy that at worst reproduces behaviour subject to the same constraints. Our algorithm is fast, robust and scales to complex high-dimensional movement systems. Furthermore it is able to deal with constraints that are both *non-linear* and *discontinuous* in time and space.

2. Problem Formulation

In this section, we characterise the problem of DPL when constraints are applied to motion, and we describe the special case of potential-based policies.

2.1. Direct Policy Learning

Following Schaal et al (2003), we consider the learning of autonomous kinematic policies

$$\dot{\mathbf{x}}(t) = \boldsymbol{\pi}(\mathbf{x}(t)) \ , \qquad \boldsymbol{\pi} : \mathbb{R}^n \mapsto \mathbb{R}^n, \tag{1}$$

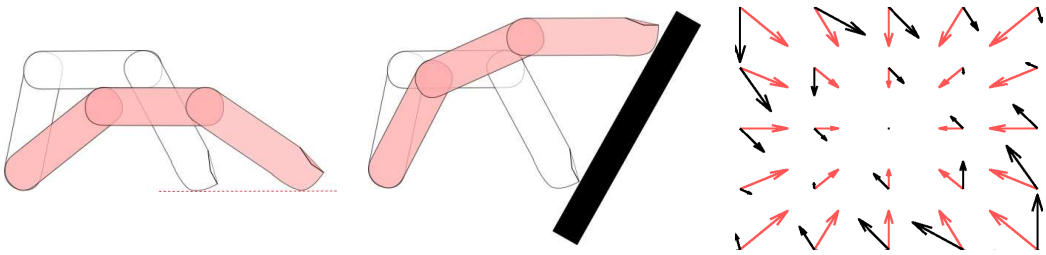
where $\mathbf{x} \in \mathbb{R}^n$ is some appropriately[1] chosen state-space and $\dot{\mathbf{x}} \in \mathbb{R}^n$ is the desired change in state. The goal of DPL is to approximate the policy (1) as closely as possible (Schaal et al 2003). It is usually formulated as a supervised learning problem where it is assumed that we have observations of $\dot{\mathbf{x}}(t)$, $\mathbf{x}(t)$ (often in the form of trajectories), and from these we wish to learn the mapping $\boldsymbol{\pi}$. In previous work this has been done by fitting parametrised models in the form of dynamical systems (Ijspeert et al 2003, 2002), non-parametric modelling (Peters and Schaal 2008; Calinon and Billard 2007; D'Souza et al 2001), probabilistic Bayesian approaches (Grimes et al 2007, 2006) and hidden Markov models (Takano et al 2006; Inamura et al 2004).

An implicit assumption found in DPL approaches to date is that the data used for training comes from behavioural observations of some *unconstrained* or *consistently constrained* policy (Calinon and Billard 2007). By this it is meant that the policy is observed either under no constraint (e.g. movements in free space such as gestures or figure drawing), or under constraints consistent over observations (e.g. interacting with the same objects or obstacles in each case). However, in many everyday behaviours, there is variability in the constraints, such as when opening doors of varying sizes or walking on uneven terrain. This *variability in the constraints* cannot be accounted for by standard DPL approaches.

2.1.1. Example: Finger Extension with Contact Constraints

As an example, consider the learning of a simple policy to extend a jointed finger. In Fig. 2(a) the finger is unconstrained and the policy simply moves the joints towards the zero (outstretched) position. On the other hand, in Fig. 2(b), an obstacle lies in the path of the finger, so that the finger movement is constrained – it is not able to

[1]It should be noted that, as with all DPL approaches, the choice of state-space is problem specific (Schaal et al 2003) and, when used for imitation learning, depends on the *correspondence* between demonstrator and imitator. For example if we wish to learn the policy a human demonstrator uses to wash a window, and transfer that behaviour to an imitator robot, an appropriate choice of $\mathbf{x}$ would be the Cartesian coordinates of the hand, which would correspond to the end-effector coordinates of the robot. Transfer of behaviour across non-isomorphic state spaces, for example if the demonstrator and imitator have different embodiments, is also possible by defining an appropriate state-action metric (Alissandrakis et al 2007).



(a) (b) (c)

Figure 2. Illustration of two apparently different behaviours from the same policy: (a) unconstrained movement (b) movement constrained by an obstacle (black box) (c) vector field visualisation of the unconstrained (red) and constrained (black) policy for two of the finger joints as a function of their angles.

penetrate the obstacle, so moves along the surface. The vector field representation of the two behaviours is shown in Fig. 2(c).

Given the task of learning in this scenario, applying traditional DPL approaches would result in one of two possibilities. The first is that if the observations are *labelled with respect to the constraint* (here, the orientation, position and shape of the obstacle) one could learn a separate policy model for the behaviour in each of the settings. However this is clearly unsatisfactory, since each model would only be valid for the specific setting, and we would need increasing numbers of models as we observed the policy under new constraints (for example different shaped obstacles at different positions and orientations). The second possibility is that the data is *unlabelled* with respect to the constraint. In this case, one might try to perform regression directly on the observations, that is observations from both vector fields (cf. Fig. 2(c), black and red vectors). However, this presents the problem that *model averaging* would occur across observations under different constraints, resulting in a poor representation of the movement in terms of the magnitude and direction of the predictions (see Sec. 2.3.).

We can avoid the need for multiple policy models if we relax our assumptions on the form (1) of the observed commands, and allow for an additional transformation of $\boldsymbol{\pi}(\mathbf{x})$. We thus model both the red and black observations as stemming from the same policy ('extend the finger'), and attribute its different appearance to the transformations as induced by the constraints. With a restriction on the class of possible transformations, as will be detailed in the next section, we can model the unconstrained policy even if we only observed constrained movements, and we can apply new constraints to adapt the policy to novel scenarios.

2.2. Constraint Model

In this paper we consider constraints which act as hard restrictions on movements available to the policy. Specifically, we consider policies subject to a set of k-dimensional ($k \leq n$) Pfaffian constraints

$$\mathbf{A}(\mathbf{x}, t)\dot{\mathbf{x}} = \mathbf{0}. \tag{2}$$

Under these constraints, the policy is projected into the nullspace of $\mathbf{A}(\mathbf{x}, t)$:

$$\dot{\mathbf{x}}(t) = \mathbf{N}(\mathbf{x}, t)\boldsymbol{\pi}(\mathbf{x}(t)) \tag{3}$$

where $\mathbf{N}(\mathbf{x},t) \equiv (\mathbf{I} - \mathbf{A}^\dagger\mathbf{A}) \in \mathbb{R}^{d\times d}$ is in general a non-linear, time-varying projection operator[1], $\mathbf{A}(\mathbf{x},t) \in \mathbb{R}^{k\times d}$ is some matrix describing the constraint and $\mathbf{I} \in \mathbb{R}^{d\times d}$ is the identity matrix. Constraints of the form (2) commonly appear in scenarios where manipulators interact with solid objects, for example when grasping a tool or turning a crank or a pedal, that is, contact constraint scenarios (Park and Khatib 2006; Murray et al 1994; Mattikalli and Khosla 1992). Such constraints are also common in the control of redundant degrees of freedom in high-dimensional manipulators (Liégeois 1977; Khatib 1987; Peters et al 2008), where policies such as (3) are used, for example, to aid joint stabilisation (Peters et al 2008), avoid joint limits (Chaumette and Marchand 2001), kinematic singularities (Yoshikawa 1985) or obstacles (Choi and Kim 2000; Khatib 1985) under task constraints. As an example: Setting $\mathbf{A}$ to the Jacobian that maps from joint-space to end-effector position coordinates would allow any motion in the joint space provided that the end-effector remained stationary. The formalism is generic and extends to a wide variety of systems; for example Antonelli et al (2005) apply it to team coordination in mobile robots and Itiki et al (1996) use the formalism to model the dynamics of jumping.

In general the effect of constraints (2)-(3) is to disallow motion in some subspace of the system, specifically the space orthogonal to the image of $\mathbf{N}(\mathbf{x},t)$. In essence these components of motion are *projected out* of the observed movement. For example, as illustrated in Fig. 3 (left), a policy $\boldsymbol{\pi}$ is constrained in two different ways corresponding to two different projections of the unconstrained movement. In the first observation $\dot{\mathbf{x}}_1$, the constraint prevents movement in the direction normal to the vertical plane[1]. For the second observation $\dot{\mathbf{x}}_2$, the constraint only allows movement in the horizontal plane.

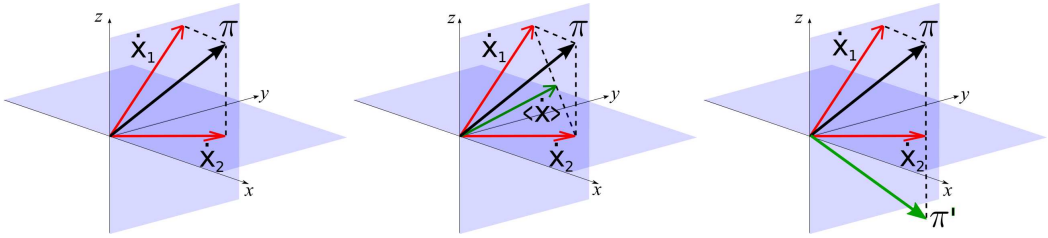


Figure 3. Illustration of the effect of constraints on the unconstrained policy, the averaging effect of standard DPL and the degeneracy problem. Left: Two constraints applied to the policy $\boldsymbol{\pi}$ result in projected observations $\dot{\mathbf{x}}_1, \dot{\mathbf{x}}_2$. Centre: direct regression results in averaging of the two movements $\langle\dot{\mathbf{x}}\rangle$ in a way that cannot explain the observations. Right: Two policies $\boldsymbol{\pi}, \boldsymbol{\pi}'$ that both may be constrained in such a way as to produce the observation $\dot{\mathbf{x}}_2$.

2.3. Learning from Constrained Motion Data

From the viewpoint of learning, constraints as described in the previous section present problems for traditional DPL approaches. Specifically there are two issues that must be dealt with; that of *non-convexity* of observations and *degeneracy* between policies (Howard et al 2008).

The *non-convexity* problem comes from the fact that between observations, or even during the course of a single observation, constraints may change. For example consider Fig. 3 (centre). There the two policy observations under the different constraints, $\dot{\mathbf{x}}_1$ and $\dot{\mathbf{x}}_2$, appear different depending on the constraint. To the learner,

[1]Here, $\mathbf{A}^\dagger$ denotes the (unweighted) Moore-Penrose pseudoinverse of the matrix $\mathbf{A}$

[1]It should be noted that in general the orientation of the constraint plane onto which the policy is projected may vary both with state position and time.

this means that the data from the two scenarios will appear *non-convex*, in the sense that for any given point in the input ($\mathbf{x}$) space multiple outputs ($\dot{\mathbf{x}}$) may exist. This causes problems for supervised learning algorithms, for example directly training on these observations may result in model-averaging. Here, averaging of $\dot{\mathbf{x}}_1, \dot{\mathbf{x}}_2$ results in the prediction $\langle \dot{\mathbf{x}} \rangle$ that clearly does not match the unconstrained policy $\boldsymbol{\pi}$, either in direction or magnitude (ref. Fig. 3, centre).

The *degeneracy* problem stems from the fact that for any given set of projected (constrained) policy observations, there exist multiple candidate policies that could have produced that movement. This is due to the projection eliminating components of the unconstrained policy that are orthogonal to the image of $\mathbf{N}(\mathbf{x}, t)$ so that the component of $\boldsymbol{\pi}$ in this direction is undetermined by the observation. For example consider the constrained observation $\dot{\mathbf{x}}_2$ in Fig. 3 (right). There motion in the y direction is restricted, meaning that that component of $\boldsymbol{\pi}$ is not seen in this observation. Given only $\dot{\mathbf{x}}_2$ we cannot determine if the policy $\boldsymbol{\pi}$ or an alternative, such as $\boldsymbol{\pi}'$ (ref. Fig. 3, right) produced the observation. In effect we are not given sufficient information about the unconstrained policy to guarantee that it is fully reconstructed.

Despite these restrictions, we wish to do the best we can with the data available. We adopt a strategy whereby we look for policies that are, as a minimum, consistent with the constrained observations $\dot{\mathbf{x}}$. For example, returning to Fig. 3, if we only observe $\dot{\mathbf{x}}_2$, (that is the policy under a single, specific constraint) the simplest (and safest) strategy would be to use that same vector as our prediction. In this way we can at least accurately predict the policy under that constraint (albeit only under that particular constraint). If we are given further observations under new constraints we can recover more information about the unconstrained policy $\boldsymbol{\pi}$. For instance, observing $\dot{\mathbf{x}}_1$ eliminates the possibility that $\boldsymbol{\pi}'$ underlies the movements since it cannot project onto both $\dot{\mathbf{x}}_1$ and $\dot{\mathbf{x}}_2$. Applying this strategy for increasing numbers of observations, our model will not only generalise over the constraints seen, but also come closer to the unconstrained policy $\boldsymbol{\pi}$.

Finally, it should be noted that if in all observations certain components of the policy are constrained, then we can never hope to uncover those components. However, in such cases it is reasonable to assume that, if these components are always eliminated by the constraints, then they are not relevant for the scenarios in which movements were recorded.

Despite the problems of learning policies under the constraints outlined recent studies (Howard et al 2006; Howard and Vijayakumar 2007; Howard et al 2008) have suggested that for the important special class of *potential-based* policies it is possible to efficiently learn the unconstrained policy. We characterise this class of policies in the next section.

2.4. Potential-based Policies

A potential-based policy is a policy defined through the gradient of a scalar potential function $\phi(\mathbf{x})$

$$\boldsymbol{\pi}(\mathbf{x}) = -\nabla_{\mathbf{x}} \phi(\mathbf{x}). \tag{4}$$

Such policies can be thought of as greedily optimising the potential function at every time step (Nakamura 1991) and thus encode attractor landscapes where the minima of the potential correspond to stable attractor points. An example is given in Fig. 4 where a potential function with three maxima (repellors) and two minima (attractors) is shown and the corresponding policy is overlaid (black vectors).

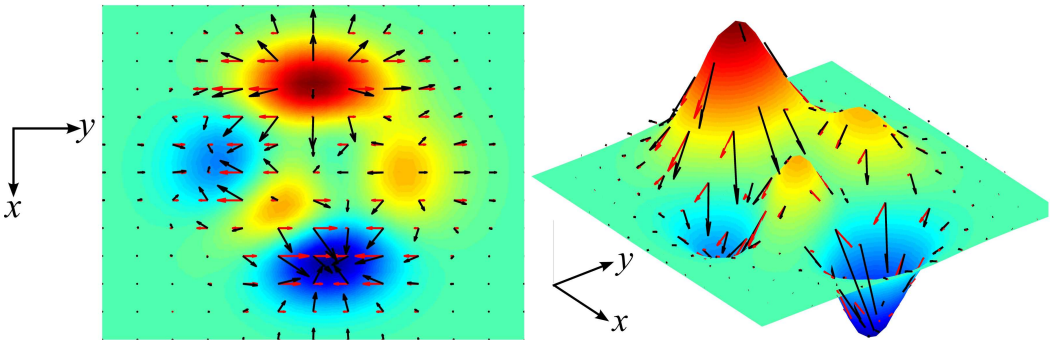


Figure 4. Potential function with three maxima (repellors) and two minima (attractors). Overlaid are the corresponding unconstrained policy vectors (black) and a set of constrained policy vectors (red).

A wide variety of behaviours may be represented as potential-based. For example, reaching behaviours may be represented by a potential defined in hand space, with a single minimum at the target. Furthermore decision-based behaviours may be encoded as potentials (Körding and Wolpert 2004; Körding et al 2004; Chajewska et al 2001, 1998). For example when reaching for an object, a potential may be defined with two minima, one corresponding to reaching with the right hand, the other reaching with the left. The decision of which hand to use for reaching would thus be determined by the start state (e.g. reach with the closest hand) and the relative offset of the two minima (e.g. right-handedness would imply a lower minimum for that hand). Potential-based policies are also extensively used as nullspace policies in control of redundant manipulators (Gienger et al 2005; English and Maciejewski 2000; Chaumette and Marchand 2001; Choi and Kim 2000; Nakamura 1991; Yoshikawa 1985), and for navigation and obstacle avoidance problems in mobile robotics (Ren et al 2006; Conner et al 2003; Rimon and Koditschek 1992). Furthermore, in reinforcement learning and optimal control (Sutton and Barto 1998; Todorov 2006), policies that are greedy with respect to the value function can be thought of as potential-based, in the sense that the policy does a gradient descent on the value function.

2.4.1. Learning from Constrained Potential-based Policies

If the policy under observation is potential-based, an elegant solution to solving the non-convexity and degeneracy problems is to model the policy's *potential function* (Howard et al 2008; Howard and Vijayakumar 2007) rather than modelling it directly. This is due to a special property of constrained potential-based policies, namely that observations of the constrained movements give us information about the shape of the underlying potential, up to a translation in ϕ corresponding to constants of integration for the observations.

In Fig. 4 this is shown for a potential function defined over a two-dimensional state-space (top and 3-D perspective views). The potential function (colours) and unconstrained policy (black vectors) is shown, along with the policy subject to a constraint (red vectors). For the case of potential-based policies the policy vectors are given by the gradient vector of the potential (as expressed in (4)). This means that the (unconstrained) policy vectors point in the direction of steepest descent, with the magnitude equal to the slope in that direction (Fig. 4, black vectors).

Now, if a constraint is applied, the direction and magnitude of the vectors change. In the example in Fig. 4 the constraint prevents movement in one dimension (x dimension in Fig. 4, left) so that only motion corresponding the the second dimension (y dimension in Fig. 4, left) is observed. The vectors now point in the direction of steepest descent *subject to the constraint*, with magnitude equal to the slope of

the potential in that direction, as can be seen from Fig. 4, right. In other words the projected vectors correspond to the *directional derivatives* of the potential, in the direction parallel to the observations.

This lends us the opportunity of modelling the unconstrained policy, by piecing together information about the slope of the potential in different directions. For each observation (e.g. $\dot{\mathbf{x}}_1$ in Fig. 3) we get information about the directional derivative in that direction (i.e. the direction parallel to $\dot{\mathbf{x}}_1$). This means we transform the problem of combining these n-dimensional vector observations (ref. Fig. 3) to one of 'piecing together' local estimates of the slope of the potential.

A convenient method for doing this is to use line integration to accurately estimate the form of the potential along trajectories (Howard et al 2008; Howard and Vijayakumar 2007) and then use these local estimates to build a global model of the potential. We outline a method for doing this in the next section.

3. Learning Nullspace Policies Through Local Model Alignment

In the following we propose a method for modelling the potential from constrained motion data. Given observations of constrained trajectories, we first model the potential on a trajectory-wise basis using numerical line integration. We then consolidate these trajectory-wise models using results from recent work in dimensionality reduction (Verbeek 2006; Verbeek et al 2004) to ensure consistency. Finally, we use these consistent models to learn a global model of the potential function, and thus the policy, for use in control.

3.1. Estimating the potential along single trajectories

As has been described in (Howard et al 2008; Howard and Vijayakumar 2007), it is possible to model the potential along sampled trajectories using a form of line integration. Specifically, combining (3) and (4), the (continuous time) state evolution of the system is given by

$$\dot{\mathbf{x}} = \mathbf{N}(\mathbf{x}, t)\boldsymbol{\pi}(\mathbf{x}) = -\mathbf{N}(\mathbf{x}, t)\nabla_{\mathbf{x}}\phi(\mathbf{x}) \tag{5}$$

Let $\bar{\mathbf{x}}(t)$ be the solution of (5). If we line-integrate along $\bar{\mathbf{x}}(t)$ we have

$$\int_{\bar{\mathbf{x}}} (\nabla_{\mathbf{x}}\phi)^T \, d\mathbf{x} = \int_{t_0}^{t_f} (\nabla_{\mathbf{x}}\phi)^T \dot{\mathbf{x}} \, dt = -\int_{t_0}^{t_f} (\nabla_{\mathbf{x}}\phi)^T \mathbf{N}(\mathbf{x}, t)\nabla_{\mathbf{x}}\phi(\mathbf{x}) \, dt, \tag{6}$$

where t_0 and t_f are the start and finishing instants of $\bar{\mathbf{x}}(t)$. We assume that we have recorded trajectories $\mathbf{x}(t), \dot{\mathbf{x}}(t)$ of length T sampled at some sampling rate $1/\delta t$ Hz so that for each trajectory we have a tuple of points $\mathbf{X}_k = \mathbf{x}_{k,1}, \ldots, \mathbf{x}_{k,T\delta t}$. Now, assuming the sampling rate to be sufficiently high, we can make a linear approximation to (5)

$$\mathbf{x}_{i+1} \approx \mathbf{x}_i + \delta t \, \mathbf{N}_i \boldsymbol{\pi}_i = \mathbf{x}_i - \delta t \, \mathbf{N}_i \nabla_{\mathbf{x}}\phi(\mathbf{x}_i) \tag{7}$$

and (6) can be approximated using an appropriate numerical integration scheme. An example of such a scheme is Euler integration, which involves the first order approximation

$$\phi(\mathbf{x}_{i+1}) \approx \phi(\mathbf{x}_i) + \frac{1}{\delta t}(\mathbf{x}_{i+1} - \mathbf{x}_i)^T \mathbf{N}_i \nabla_{\mathbf{x}}\phi(\mathbf{x}_i). \tag{8}$$

Since the effect of the time constant δt is simply to scale the discretised policy vectors, we can neglect it by scaling time units such that $\delta t = 1$. This comes with the proviso that for implementation on the imitator robot, the learnt policy may need to be scaled back to ensure that the correct time correspondence is kept. For steps $\mathbf{x}_i \to \mathbf{x}_{i+1}$ that follow the projected policy (3) we can rearrange (7) with the scaled time coordinates, and substitute into (8) to yield

$$\phi(\mathbf{x}_{i+1}) \approx \phi(\mathbf{x}_i) - \|\mathbf{x}_{i+1} - \mathbf{x}_i\|^2, \tag{9}$$

where the negative sign reflects our assumption (as expressed in (4)) that attractors are minima of the potential. We use this approximation to generate estimates $\hat{\phi}(\mathbf{x}_i)$ of the potential along any given trajectory $\mathbf{x}_1, \mathbf{x}_2 \dots \mathbf{x}_N$ in the following way: We set $\hat{\phi}_1 = \hat{\phi}(\mathbf{x}_1)$ to an arbitrary value and then iteratively assign $\hat{\phi}_{i+1} := \hat{\phi}_i - \|\mathbf{x}_{i+1} - \mathbf{x}_i\|^2$ for the remaining points in the trajectory.

Note that an arbitrary constant can be added to the potential function without changing the policy. Therefore, 'local' potentials that we estimate along different trajectories need to be *aligned* in a way that their function value matches in intersecting regions. We'll turn to this problem in the next section.

3.2. Constructing the global potential function

Let us assume we are given K trajectories $\mathbf{X}_k = (\mathbf{x}_{k1}, \mathbf{x}_{k2} \dots \mathbf{x}_{kN_k})$ and corresponding point-wise estimates $\hat{\mathbf{\Phi}}_k = (\hat{\phi}_{k1}, \hat{\phi}_{k2} \dots \hat{\phi}_{kN_k})$ of the potential, as provided from the Euler integration just described. In a first step, we fit a function model $f_k(\mathbf{x})$ of the potential to each tuple $(\mathbf{X}_k, \hat{\mathbf{\Phi}}_k)$, such that $f_k(\mathbf{x}_i) \approx \hat{\phi}_{ki}$. Although in principle any regression method could be applied here, our options are somewhat limited by the fact that these possibly non-linear models have to be acquired from the few data points available in each trajectory. To avoid unnecessary complications, we choose a nearest-neighbour (NN) regression model, i.e.,

$$f_k(\mathbf{x}) = \Phi_{ki^*} \quad , \quad i^* = \arg\min_i \|\mathbf{x} - \mathbf{x}_{ki}\|^2. \tag{10}$$

Since we wish to combine the models to a global potential function, we need to define some function for weighting the outputs of the different models. For the NN algorithm, we choose to use a Gaussian kernel

$$w_k(\mathbf{x}) = \exp\left[-\frac{1}{2\sigma^2} \min_i \|\mathbf{x} - \mathbf{x}_{ki}\|^2\right]. \tag{11}$$

From these weights we can calculate responsibilities

$$q_k(\mathbf{x}) = \frac{w_k(\mathbf{x})}{\sum_{i=1}^K w_i(\mathbf{x})} \tag{12}$$

and a (naive) global prediction $f(\mathbf{x}) = \sum_{k=1}^K q_k(\mathbf{x}) f_k(\mathbf{x})$ of the potential at $\mathbf{x}$. However, as already stated, the potential is only defined up to an additive constant, and most importantly this constant can vary from one local model to another. This means that we first have to shift the models by adding some *offset* to their estimates of the potential, such that all local models are *in good agreement* about the global potential at any number of states $\mathbf{x}$.

Fortunately, a similar problem has already been tackled in the literature: In the field of non-linear dimensionality reduction, Verbeek et al (2004) have shown

how to align multiple local PCA models into a common low-dimensional space. In particular, they endowed each local PCA model with an additional affine mapping $\mathbf{g}_k(\mathbf{z}) = \mathbf{A}_k\mathbf{z}+\mathbf{b}_k$, which transformed the coordinates $\mathbf{z}_k$ of a data point *within* the k-th PCA model into the desired global coordinate system. Verbeek et al (2004) retrieved the parameters of the optimal mappings $\mathbf{g}_k$ by minimising the objective function

$$E = \frac{1}{2}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{j=1}^{K} q_{km}q_{jm}\|\mathbf{g}_{km} - \mathbf{g}_{jm}\|^2, \tag{13}$$

where $\mathbf{g}_{km}$ denotes the coordinate of the m-th data vector, as mapped through the k-th PCA model, and q_{km} is the corresponding responsibility of that model. The objective can easily be interpreted as the 'disagreement' between any two models, summed up over all data points, and weighted by the responsibilities of two models each. That is, the disagreement for any combination of m, k and j only really counts, if the responsibility of both the k-th and the j-th model is sufficiently high for the particular query point. Notably, E is convex and can be minimised by solving a generalised eigenvalue problem of moderate dimensions, that is, there are no local minima, and the solution can be found efficiently.

In analogy to the PCA-alignment method (Verbeek et al 2004), we augment our local potential models $f_k(\cdot)$ by a scalar offset b_k and define the corresponding objective function as

$$\begin{aligned} E(b_1 \dots b_K) = \frac{1}{2}\sum_{m=1}^{M}\sum_{k=1}^{K}\sum_{j=1}^{K} q_k(\mathbf{x}_m)q_j(\mathbf{x}_m) \times \\ \left((f_k(\mathbf{x}_m) + b_k) - (f_j(\mathbf{x}_m) + b_j)\right)^2, \end{aligned} \tag{14}$$

or, in a slightly shorter form,

$$E(\mathbf{b}) = \frac{1}{2}\sum_{m,k,j} q_{km}q_{jm}\left(f_{km} + b_k - f_{jm} - b_j\right)^2. \tag{15}$$

Here, $\sum_m$ denotes a summation over the complete data set, that is, over all points from all trajectories ($M = \sum_{k=1}^{K} N_k$). Using the symmetry in $j \leftrightarrow k$ and $\sum_k q_{kn} = 1$, we split (15) into terms that are constant, linear, or quadratic in b_k, yielding

$$\begin{aligned} E(\mathbf{b}) &= \sum_{m,k} q_{km}f_{km}^2 - \sum_{m,j,k} q_{km}q_{jm}f_{km}f_{jm} \\ &\quad +2\sum_{m,k} q_{km}f_{km}b_k - 2\sum_{m,k} q_{km}q_{jm}f_{jm}b_k \\ &\quad +\sum_{m,k} q_{km}b_k^2 - \sum_{m,k,j} q_{km}q_{jm}b_kb_j \\ &= E_0 + 2\mathbf{a}^T\mathbf{b} + \mathbf{b}^T\mathbf{H}\mathbf{b}. \end{aligned} \tag{16}$$

Here, we introduced E_0 as a shortcut for the terms independent of $\mathbf{b}$, the vector $\mathbf{a} \in \mathbb{R}^K$ with elements $a_k = \sum_m q_{km}f_{km} - \sum_{m,j} q_{km}q_{jm}f_{jm}$, and the Hessian matrix $\mathbf{H} \in \mathbb{R}^{K\times K}$ with elements $h_{ij} = \delta_{ij}\sum_m q_{jm} - \sum_m q_{im}q_{jm}$. The objective function

is quadratic in $\mathbf{b}$, so we retrieve the optimal solution by setting the derivatives to zero, which yields the equation $\mathbf{Hb} = -\mathbf{a}$.

However, note that a common shift of all offsets b_k does not change the objective (14), which corresponds to the shift-invariance of the global potential. Therefore, the vector $(1, 1, \ldots, 1)^T$ spans the nullspace of $\mathbf{H}$, and we need to use the pseudo-inverse of $\mathbf{H}$ to calculate the optimal offset vector

$$\mathbf{b}_{opt} = -\mathbf{H}^{\dagger}\mathbf{a}. \tag{17}$$

Compared to aligning PCA models, the case we handle here is simpler in the sense that we only need to optimise for scalar offsets b_k instead of affine mappings. On the other hand, our local potential models are non-linear, have to be estimated from relatively little data, and therefore do not extrapolate well, as will be discussed in the following section.

3.3. Smoothing parameter selection and outlier detection

Since we restrict ourselves to using simple NN regression for the local potential models in this paper, the only open parameter of our algorithm is σ^2, i.e., the kernel parameter used for calculating the responsibilities (11). A too large choice of this parameter will over-smooth the potential, because the NN regression model basically predicts a locally constant potential, but at the same time trajectories will have relatively high responsibilities for even far apart points $\mathbf{x}$ in state space.

On the other hand, a too small value of σ^2 might lead to *weakly connected trajectories*: If a particular trajectory does not make any close approach to other trajectories in the set, the quick drop-off of its responsibility implies that it will not contribute to the alignment error (based on pairs of significant responsibility), which in turn implies that its own alignment – the value of its offset – does not matter much.

The same reasoning applies to groups of trajectories that are close to each other, but have little connection to the rest of the set. For the remainder of the paper, we will refer to such trajectories as 'outliers', since like in classical statistics we need to remove these from the training set: If their influence on the overall alignment is negligible, their own alignment can be poor, and this becomes a problem when using the output of the optimisation (17) to learn a global model of the potential. To avoid interference, we only include trajectories if we are sure that their offset is consistent with the rest of the data[1].

Fortunately, outliers in this sense can be detected automatically by looking for small eigenvalues of $\mathbf{H}$: In the same way as adding the same offset to all trajectories leads to a zero eigenvalue, further very small eigenvalues and the corresponding eigenvectors indicate indifference towards a shift of some subset of trajectories versus the rest of the set. In practice, we look for eigenvalues $\lambda < 10^{-8}$, and use a recursive bi-partitioning algorithm in a way that is very similar to spectral clustering (Kannan et al 2004). We then discard all trajectories apart from those in the largest 'connected' group. Please refer to Appendix B for details of this step.

Finally, with these considerations in mind, we select the smoothing parameter σ^2 to match the scale of typical distances in the data sets. In all of the experiments

[1]It should be noted that these trajectories are not outliers in the sense of containing corrupt data and could in fact be used for further training of the model. For example one could take a hierarchical approach, where groups of strongly connected trajectories are aligned first to form models consisting of groups of trajectories with good alignment. We can then recursively repeat the process, aligning these larger (but more weakly connected) groups until all of the data has been included.

presented in this paper we used the same heuristic selection. In particular, we first calculated the distances between any two trajectories $k, j \in \{1 \dots K\}$ in the set as the distances between their closest points

$$d_{kj} = \min \left\{ \|\mathbf{x}_{kn} - \mathbf{x}_{jm}\|^2 \mid n, m \in \{1 \dots N\} \right\}, \tag{18}$$

and also the distances to the closest trajectory

$$d_k^{min} = \min \left\{ d_{kj} \mid j \neq k \right\}. \tag{19}$$

We then consider three choices for σ^2, which we refer to as 'narrow', 'wide' and 'medium':

$$\sigma^2_{nar} = \text{median} \left\{ d_k^{min} \mid k \in \{1 \dots K\} \right\} \tag{20}$$

$$\sigma^2_{wid} = \text{median} \left\{ d_{jk} \mid j, k \in \{1 \dots K\}, j \neq k \right\} \tag{21}$$

$$\sigma^2_{med} = \sqrt{\sigma^2_{nar} \sigma^2_{wid}}. \tag{22}$$

In Section 4.1. we give a comparison of the learning performance for each of these choices of σ^2 for policies of varying complexity.

3.4. Learning the global model

After calculating optimal offsets $\mathbf{b}_{opt}$ and cleaning the data set from outliers, we can learn a global model $f(\mathbf{x})$ of the potential using any regression algorithm. Here, we choose Locally Weighted Projection Regression (LWPR) (Vijayakumar et al 2005) because it has been demonstrated to perform well in cases where the data lies on low-dimensional manifolds in a high-dimensional space, which matches our problem of learning the potential from a set of trajectories. As the training data for LWPR, we use all non-outlier trajectories and their estimated potentials as given by the Euler integration *plus* their optimal offset, that is, the input-output tuples

$$\left\{ (\mathbf{x}_{kn}, \hat{\phi}_{kn} + b_k^{opt}) \mid k \in \mathcal{K}, n \in \{1 \dots N_k\} \right\}, \tag{23}$$

where $\mathcal{K}$ denotes the set of indices of non-outlier trajectories. Once we have learnt the model $f(\mathbf{x})$ of the potential, we can take derivatives to estimate the unconstrained policy $\hat{\boldsymbol{\pi}}(\mathbf{x}) = -\nabla_{\mathbf{x}} f(\mathbf{x})$. For convenience, the complete procedure is summarised in Algorithm **1**.

4. Experiments

To explore the performance of our algorithm, we performed experiments on data from autonomous kinematic control policies (Schaal et al 2003) applied[1] to different plants, including the whole body motion controller (WBM) of the humanoid robot ASIMO (Gienger et al 2005). In this section, we first discuss results from an artificial toy problem controlled according to the same generic framework to illustrate

[1]Since the goal of the experiments was to validate the proposed approach, we used policies known in closed form as a ground truth. In the follow-up paper we apply our method to human motion capture data.

Algorithm 1 PolicyAlign

1: Estimate $\mathbf{X}_k$, $\hat{\mathbf{\Phi}}_k, \{k = 1 \dots K\}$ using Euler integration. Calculate σ^2.

2: Alignment:

- Calculate prediction and responsibility of each local model f_k on each data point $\mathbf{x}_m,\ m = 1 \dots M$:
  $f_{km} = f_k(\mathbf{x}_m);\ \ q_{km} = w_k(\mathbf{x}_m) / \sum_i w_i(\mathbf{x}_m)$
- Construct $\mathbf{H}, \mathbf{a}$ with elements
  $h_{ij} = \delta_{ij} \sum_m q_{jm} - \sum_m q_{im} q_{jm};\ \ a_k = \sum_m q_{km} f_{km} - \sum_{m,j} q_{km} q_{jm} f_{jm}$
- Find optimal offsets $\mathbf{b}_{opt} = -\mathbf{H}^{\dagger}\mathbf{a}$

3: Discard outliers ($\mathbf{H}$ eigenvalues, $\lambda < 10^{-8}$).

4: Train global model on data tuples $(\mathbf{x}_{kn}, \hat{\phi}_{kn} + b_k^{opt})$

the key concepts. We then discuss an example scenario in which the algorithm is used to enable ASIMO to learn a realistic bi-manual grasping task from observations from a constrained demonstrator. We then give a brief discussion on how our algorithm scales to policies in very high dimensional systems such as for 22 DOF of the ASIMO WBM controller (Gienger et al 2005). Finally, we report the performance of the algorithm when learning from data containing a set of pathological constraints.

4.1. Toy Example

The toy example consists of a two-dimensional system with a policy defined by a quadratic potential, subject to discontinuously switching constraints. Specifically, the potential is given by

$$\phi(\mathbf{x}) = (\mathbf{x} - \mathbf{x}_c)^T \mathbf{W} (\mathbf{x} - \mathbf{x}_c) \tag{24}$$

where $\mathbf{W}$ is some square weighting matrix which we set to $0.05\mathbf{I}$ and $\mathbf{x}_c$ is a vector defining the location of the attractor point, here chosen to be $\mathbf{x}_c = \mathbf{0}$. Data was collected by recording trajectories generated by the policy from a start state distribution X_0. During the trajectories the policy was subjected to random constraints

$$\mathbf{A}(\mathbf{x}, t) = (\alpha_1, \alpha_2) \equiv \boldsymbol{\alpha} \tag{25}$$

where the $\alpha_{1,2}$ were drawn from a normal distribution, $\alpha_i = N(0, 1)$. The constraints mean that motion is constrained in the direction orthogonal to the vector $\boldsymbol{\alpha}$ in state space. To increase the complexity of the problem, the constraints were randomly switched during trajectories by re-sampling $\boldsymbol{\alpha}$ twice at regular intervals during the trajectory. This switches the direction in which motion is constrained as can be seen by sharp turns in the trajectories.

Figure 5 shows an example of our algorithm at work for a set of $K = 40$ trajectories of length $N = 40$ for the toy system. The raw data as a set of trajectories through the two-dimensional state space is shown in panel (a), whereas panel (b) additionally depicts the local potential models as estimated from the Euler integration prior to alignment. Each local model has an arbitrary offset against the true potential so there are inconsistencies between the predictions from each local model. Figure 5(c) shows the trajectories after alignment, already revealing the

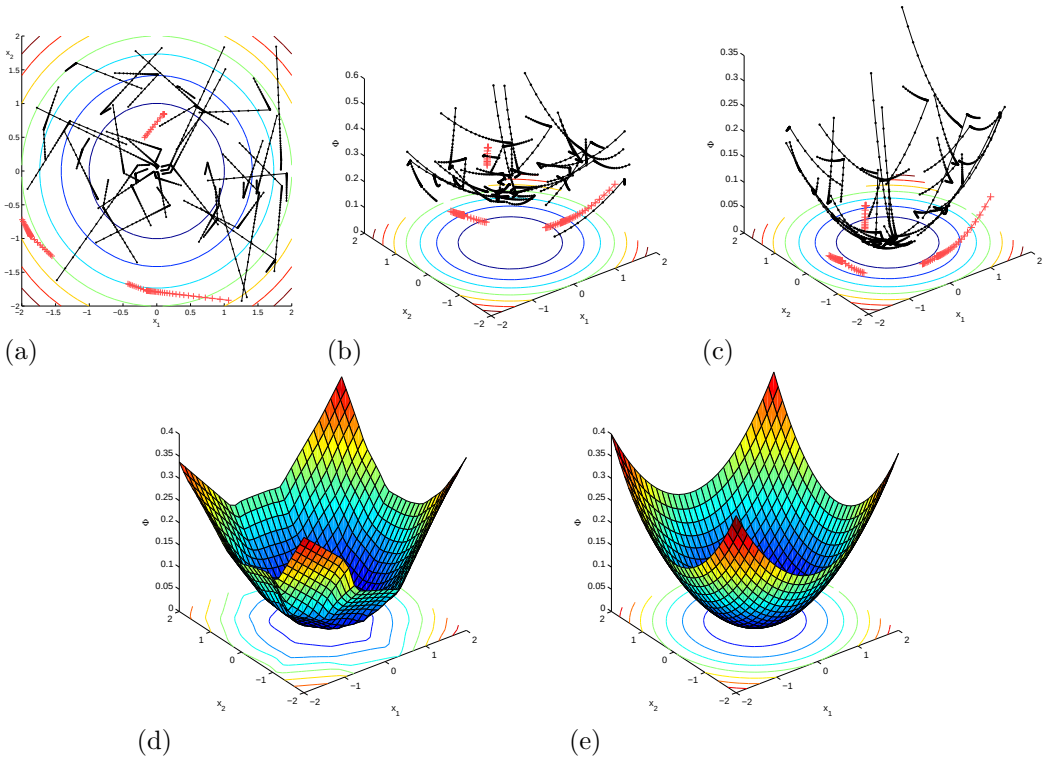


Figure 5. Top: (a) Toy data (trajectories (2-D) and contour of true potential. Estimated potential along the trajectories before (b) and after (c) alignment. Trajectories detected as difficult to align 'outliers' are shown by light crosses. Bottom: Learnt (d) and true (e) potential function after training on the aligned trajectories.

structure of the parabola.

At this point, the outlier detection scheme has identified three trajectories as being weakly connected to the remaining set. In Fig. 5(a) we can see that the outliers are indeed the only trajectories that do not have any intersection with neighbouring trajectories. At the 'narrow' length scale determined by the smoothing parameter (20), they are hard to align properly, and need to be discarded before learning the global model. Finally, Fig. 5(d) shows the global model $f(\mathbf{x})$ of the potential that was trained on the aligned trajectories, which is clearly a good approximation of the true parabolic potential shown in Fig. 5(e).

For a more thorough evaluation, we repeated this experiment on 100 data sets and evaluated[1]

- the nMSE of the aligned potential, which measures the difference between $\hat{\phi}_{kn}+b_k$ and the true potential ϕ,
- the nMSE of the learnt potential, measuring the difference between $f(\cdot)$ and $\phi(\cdot)$,
- the normalised unconstrained policy error (nUPE), quantifying the difference between $\hat{\boldsymbol{\pi}}=\nabla f$ and $\boldsymbol{\pi}=\nabla\phi$,
- the normalised constrained policy error (nCPE), which is the discrepancy between $\mathbf{N}\hat{\boldsymbol{\pi}}$ and $\mathbf{N}\boldsymbol{\pi}$, and finally
- the percentage of trajectories discarded as outliers

on a subsample of the data held out for testing. We did so for our three different choices of σ^2 given in (20-22). We also repeated the experiment using a sinusoidal

[1]A detailed explanation of the error measures used can be found in Appendix A.

potential function

$$\phi_s(\mathbf{x}) = 0.1 \sin(x_1) \cos(x_2) \tag{26}$$

with the same amount of data, as well as while using $K = 100$ trajectories of length $N = 100$ for each data set.

Table 1 summarises the results. Firstly, we can see that the 'wide' choice for σ^2 leads to large error values which are due to over-smoothing. Using the narrow σ^2, we retrieve very small errors at the cost of discarding quite a lot of trajectories[1], and the medium choice seems to strike a reasonable balance especially with respect to the nUPE and nCPE statistics. Further to this, the left panel of Fig. 6 depicts how the nUPE and nCPE evolve with increasing size of the training set, showing a smooth decline (please note the log. scale).

Secondly, when comparing the results for the parabolic and sinusoidal potentials, we can see that the latter, more complex potential (with multiple sinks) requires much more data. With only 40 trajectories and 40 points each, most of the data sets are too disrupted to learn a reasonable potential model. While at the narrow length scale (4th row), on average more than half of the data set is discarded, even the medium length scale (5th row) over-smooths the subtleties of the underlying potential.

Finally, the nCPE is always much lower than the nUPE, which follows naturally when training on data containing those very movement constraints. Still, with a reasonable amount of data, even the unconstrained policy can be modelled with remarkable accuracy.

As a final test, we also performed experiments to assess the noise robustness of the proposed approach. For this we again used data from the quadratic potential and but this time contaminated the observed states $\mathbf{x}_n$ with Gaussian noise, the scale of which we varied to match up to 20% of the scale of the data. The resulting nUPE roughly follows the noise level, as is plotted in Fig. 6 (right).

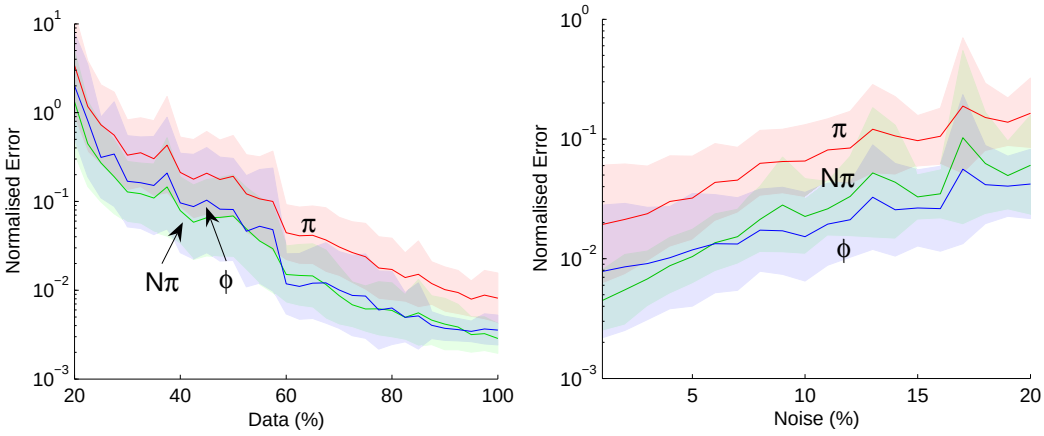


Figure 6. Learning performance on the quadratic potential (24) with varying data set sizes and noise levels. Left: Potential nMSE, nCPE and nUPE versus data set size as a percentage of the full $K = 40$ trajectories of length $N = 40$. Right: Potential nMSE, nCPE and nUPE for increasing noise levels in the observed $\mathbf{x}_n$.

[1]Please note that we also discard the outliers for evaluating the error statistics – we can hardly expect to observe good performance in regions where the learnt model $f(\mathbf{x})$ has seen no data.

Table 1. Error and outlier statistics (mean±std.dev. over 100 data sets) for the experiment on 2-D toy data. For brevity, we did not include the figures for the alignment nMSE. These were only marginally different from the potential nMSE.

Setup	σ^2	Potential nMSE	nUPE	nCPE	Disc. (%)
Parabola	narrow	0.0052 ± 0.0024	0.0486 ± 0.0211	0.0235 ± 0.0092	17.55 ± 15.96
$K = 40$	medium	0.0195 ± 0.0203	0.0859 ± 0.0486	0.0224 ± 0.0074	0.48 ± 1.11
$N = 40$	wide	0.3143 ± 0.1045	0.5758 ± 0.2726	0.1135 ± 0.0371	0 ± 0
Sinusoidal	narrow	0.0026 ± 0.0019	0.1275 ± 0.1125	0.0535 ± 0.0353	50.18 ± 14.37
$K = 40$	medium	0.0522 ± 0.0645	0.1399 ± 0.0422	0.0376 ± 0.0097	1.03 ± 3.99
$N = 40$	wide	0.5670 ± 0.1363	0.8373 ± 0.2188	0.2464 ± 0.0638	0 ± 0
Sinusoidal	narrow	0.0014 ± 0.0004	0.0657 ± 0.0142	0.0308 ± 0.0065	25.46 ± 11.42
$K = 100$	medium	0.0019 ± 0.0017	0.0628 ± 0.0089	0.0284 ± 0.0044	1.25 ± 3.33
$N = 100$	wide	0.2137 ± 0.1000	0.4262 ± 0.1367	0.1554 ± 0.0483	0 ± 0

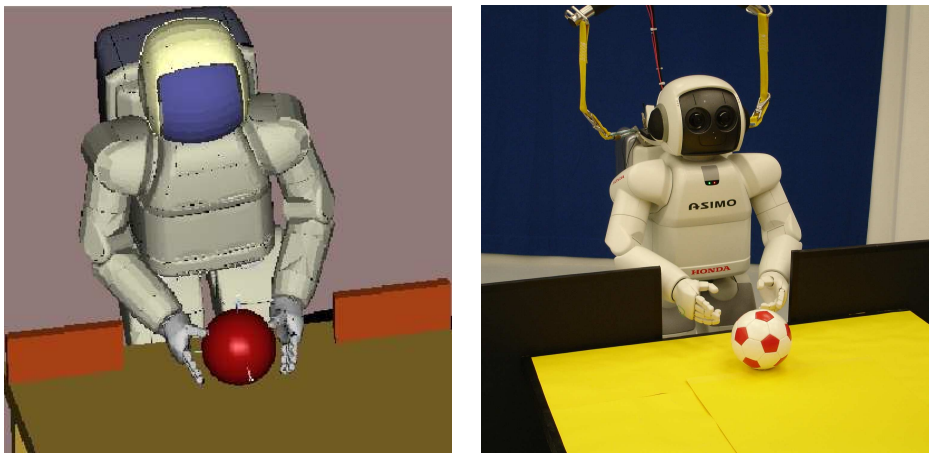


Figure 7. Example constrained reaching movement demonstrated by the expert policy. Starting with hands at the sides, the teacher robot reaches between the barriers to grasp the ball.

4.2. Grasping a Ball

The two goals of our second set of experiments were (i) to characterise how well the algorithm scaled to more complex, realistic constraints and policies and (ii) to assess how well the learnt policies generalised over different constraints. For this we set up a demo scenario in which a set of trajectories demonstrating the task of reaching for a ball on a table were given. Furthermore, it was assumed that trajectories were recorded under different contexts where different constraints applied. The goal was then to uncover a policy that both accurately reproduced the demonstrated behaviour and furthermore *generalised* to novel contexts with unseen constraints.

For this, we set up an 'expert' demonstrator from which observations were recorded. For ease of comparison with the 2-D system, the expert's policy was defined by the same quadratic potential (24) this time with the target point $\mathbf{x}_c$ corresponding to a grasping position, with the two hands positioned on either side of the ball. The state-space of the policy was defined as the Cartesian position of the two hands, corresponding to 6 DOFs[1] (hereafter, the 'task space'). In order to realise the task space policy motion, joint-space control was performed using a the ASIMO WBM controller (see Gienger et al (2005) for details).

The policy was constrained by placing a barrier on the table between the robot and the ball with a gap in it. The constraints acted on each of the hands so that motion in the direction normal to the barrier surface was prevented if a hand came too close (cf. (Sugiura et al 2007)). The constraints were such that the robot had to

[1] 3 DOFs per hand × 2 hands.

reach through the gap in order to get the ball. Such state-dependent constraints are both nonlinear in the state space and have discontinuously switching dimensionality when either one or both of the hands approach or recede from the barrier.

Data was collected by recording $K = 100$ trajectories of length $2s$ at 50 Hz, (i.e. $N = 100$ points per trajectory). Start states were sampled from a Gaussian distribution over joint configurations $\mathbf{q} \sim N(\mathbf{q}_0, 0.1\mathbf{I})$ (where $\mathbf{q}_0$ corresponds to the default standing position) and using forward kinematics to calculate the corresponding hand positions. The joint vector $\mathbf{q}$ was clipped where necessary to avoid joint limits and self collisions, and to ensure the start postures looked natural.

The constraints were varied by randomly changing the width of the gap for each trajectory. The gap widths were sampled from a Gaussian distribution $d_{gap} \sim N(\mu_{gap}, \sigma_{gap})$ where $\mu_{gap} = 0.25m$, $\sigma_{gap} = 0.1m$ and the diameter of the ball was $0.15m$. Fig. 7 shows the experimental set-up.

We used our algorithm to perform learning on 50 such data sets using the 'narrow' choice of smoothing parameter σ^2. For comparison, we also repeated the experiment on the same data, using a naive approach that learnt $\tilde{\boldsymbol{\pi}}_{naive} : \mathbf{x} \rightarrow \dot{\mathbf{x}} \in \mathbb{R}^n \mapsto \mathbb{R}^n$ by training directly on the tuples $(\mathbf{x}_i, \dot{\mathbf{x}}_i), i = 1, \ldots K \times N$ and used LWPR to learn the global model. This is in contrast to the proposed alignment scheme where we learn the 1-dimensional potential function and use the gradient of the learnt function as the policy prediction.

For this task, our algorithm achieved a very low alignment error of $6.95 \pm 0.09 \times 10^{-4}$, with $0.48 \pm 0.84\%$ of the trajectories discarded, resulting in an nMSE in the learnt potential of $7.85 \pm 0.56 \times 10^{-4}$ (mean±s.d. over 50 data sets). In Table 2 we give the errors in predicting the policy subject to (i) the training data constraints (nCPE), (ii) no constraints (nUPE), and (iii) a novel constraint, unseen in the training data. For the latter, a barrier was placed centrally between the robot and the ball, so that the robot had to reach around the barrier to grasp the ball.

The remarkably low alignment error can be attributed to the fact that in most of the observations the grasping task was achieved successfully despite the constraints forcing the hands to take alternative routes to the ball. This meant many of the trajectories closely approached the minimum of the potential, making the alignment easier around this point. This is further indicated by the low percentage of trajectories discarded.

The key result, however, can be seen by examining the policy errors (ref. Table 2). Comparing the two approaches, both achieve a similar nCPE, with the naive approach in fact performing slightly better. This indicates that the two methods both do equally well in modelling the constrained movement observations to approximately the same level of accuracy.

However, when comparing the errors for the unconstrained policy, and the policy subject to the unseen constraint, a different picture emerges. Using the model learnt by the alignment approach, the unconstrained policy predictions, and the predictions under the unseen constraint, maintain a similar level of error to that of the constrained policy. However, in stark contrast to this, the naive approach fares very poorly, with a large jump in error when predicting the policy under the new barrier constraint and when predicting the unconstrained behaviour.

The difference in the two approaches is highlighted if we compare trajectories generated by the two policies. In Fig. 8 we show example trajectories for the unconstrained reaching movement produced by the expert (black), and the policies learnt by (i) the naive approach (green), and (ii) the alignment approach (red). In the former the hands take a curved path to the ball, reproducing the average behaviour of the demonstrated (constrained) trajectories – the naive method is unable to extract the underlying task (policy) from the observed paths around the

obstacles. Consequently, it cannot generalise and find its way around the unseen barrier. In contrast, the policy learnt with the alignment approach better predicts the unconstrained policy, enabling it to take a direct route to the ball that closely matches that of the expert.

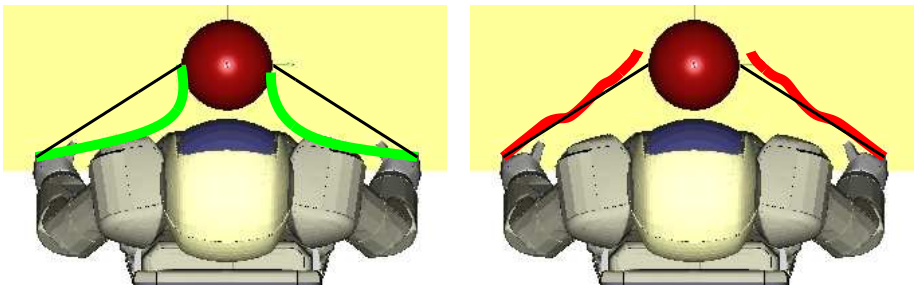


Figure 8. Unconstrained reaching movement for the expert policy (black), the policy learnt with the naive approach (green) and that learnt with the policy alignment algorithm (red).

Table 2. Constrained policy nMSE for unseen constraints on the ball-grasping task. Values are mean±s.d. over 50 data sets.

Constraint	Naive	PolicyAlign
Training	0.1298 ± 0.0113	0.1691 ± 0.0289
Unseen Barrier	0.5108 ± 0.0327	0.2104 ± 0.0357
Unconstrained	0.8766 ± 0.0589	0.2277 ± 0.0386

4.3. Learning from high-dimensional ASIMO data

In our next set of experiments we tested the scalability of our approach for learning in very high dimensions. For this we again used the quadratic potential (24) where now the state vector $\mathbf{x}$ corresponded to the 22-dimensional joint configuration of the upper body of the ASIMO humanoid robot (ref. Fig. 1). In this experiment, the policy was constrained such that in each trajectory one of the hands of the robot was constrained to lie in a plane of random orientation. Such constraints occur in a variety of behaviours where contact must be maintained with a surface, for example when writing on a whiteboard or when wiping a window (Park and Khatib 2006).

We ran the experiment on 50 data sets of $K = 100$ trajectories of length $N = 100$, with start states selected using the same process as described in the preceding section. Using the narrow setting of the smoothing parameter the algorithm achieved an alignment error of $1.6 \pm 0.3 \times 10^{-3}$ with just $0.02 \pm 0.14\%$ of the trajectories discarded. Learning on this data with LWPR, we achieved an nMSE in the learnt potential of $1.5 \pm 0.4 \times 10^{-3}$, nCPE of 0.065 ± 0.014 and nUPE of 0.157 ± 0.047. We consider this to be remarkably good performance given the high dimensionality of the input space and the relatively small size of the data set.

4.4. Pathological Constraints

In our final set of experiments we briefly characterise the performance of the algorithm subject to pathological constraints in the data. As an illustrative example, we simulated a constrained planar three-link arm, with revolute joints and unit link lengths.

The experimental set up was as follows. Data was collected from the arm moving according to the quadratic potential (24) (with $\mathbf{x}_c = \mathbf{0}$ and $\mathbf{W} = 0.05\mathbf{I}$) from a random distribution of start states. The movement of the arm was restricted by constraining the end-effector to move along a line. Mathematically the constraint matrix was

$$\mathbf{A}(\mathbf{x}, t) = \hat{\mathbf{n}}^T \mathbf{J}_{hand}(\mathbf{x}) \tag{27}$$

where $\hat{\mathbf{n}}$ is a unit vector normal to the hand-space plane and $\mathbf{J}_{hand}(\mathbf{x})$ is the hand Jacobian. The constraint was varied by altering the orientation of the plane by drawing $\hat{\mathbf{n}}$ from a uniform random distribution $U_{\hat{\mathbf{n}}}$ at the start of each trajectory.

We ran experiments on 50 such data sets each containing $K = 100$ trajectories of length $N = 100$. For this learning problem, the algorithm achieved nUPE of 0.3524 ± 0.1626 and nCPE of 0.0455 ± 0.0276. The nMSE in the learnt potential was 0.1739 ± 0.1424 with $10.28 \pm 8.25\%$ trajectories discarded. In comparison the naive approach to learning achieved nUPE of 0.8008 ± 0.0274 and nCPE of 0.0105 ± 0.0023.

The reason for the comparatively high nUPE here becomes clear if we analyse the effect of the constraints on the movement of the arm (see Fig. 9). In Fig. 9(a) the training data trajectories are plotted over the three joints of the arm. It can be seen that the trajectories do not reach the point attractor at $\mathbf{x} = \mathbf{0}$, but rather move to a line in joint space (shown in black). This 'line attractor' represents the minimum of the potential that can be reached without breaking the constraints. No trajectories travel in the direction parallel to this line. Furthermore, away from this line there are few points where trajectories come close to one another or intersect. The effect of this is that the algorithm gets little or no information about how the potential changes in the direction parallel to the line.

This is confirmed by comparing how the nUPE and nCPE change as we move along the line attractor, and radially outward from it. In Fig. 9 we show the potential nMSE, nUPE and nCPE on data contained within different regions of the state space.

Firstly, we evaluated the error on data points contained between two planes normal to the line attractor at distance d from the point attractor $\mathbf{x} = \mathbf{0}$ (Fig 9(b), dashed lines), and plotted it with increasing d (Fig 9(d)). We can see that close to $\mathbf{x} = \mathbf{0}$, the potential nMSE and nUPE start low but increase rapidly for large d. On the other hand the nCPE stays approximately constant over the entire set.

Secondly, we looked at how the errors change as we move radially outward. For this we evaluated errors on data contained within a cylinder of radius r centred on the line attractor (Fig 9(c), dashed lines). Fig 9(e) shows the change in error with increasing radius r. Again the nCPE remains constant. This time, however, the potential nMSE and nUPE are high even at small r. This indicates that the points at the two ends of the line are contributing most of the error.

Clearly in this example, the adverse constraints in the training data prevent our algorithm from fully reconstructing the unconstrained policy. The constraints prevent motion parallel to the line attractor so we cannot recover the form of the potential along that direction. However, the good performance in terms of the nCPE indicates that, at the very least, the algorithm is able to reconstruct the policy under the same constraints despite these adverse conditions.

5. Conclusion

We have proposed a novel approach to direct learning of potential-based policies from constrained motion data. Our method is fast and data-efficient, and it scales

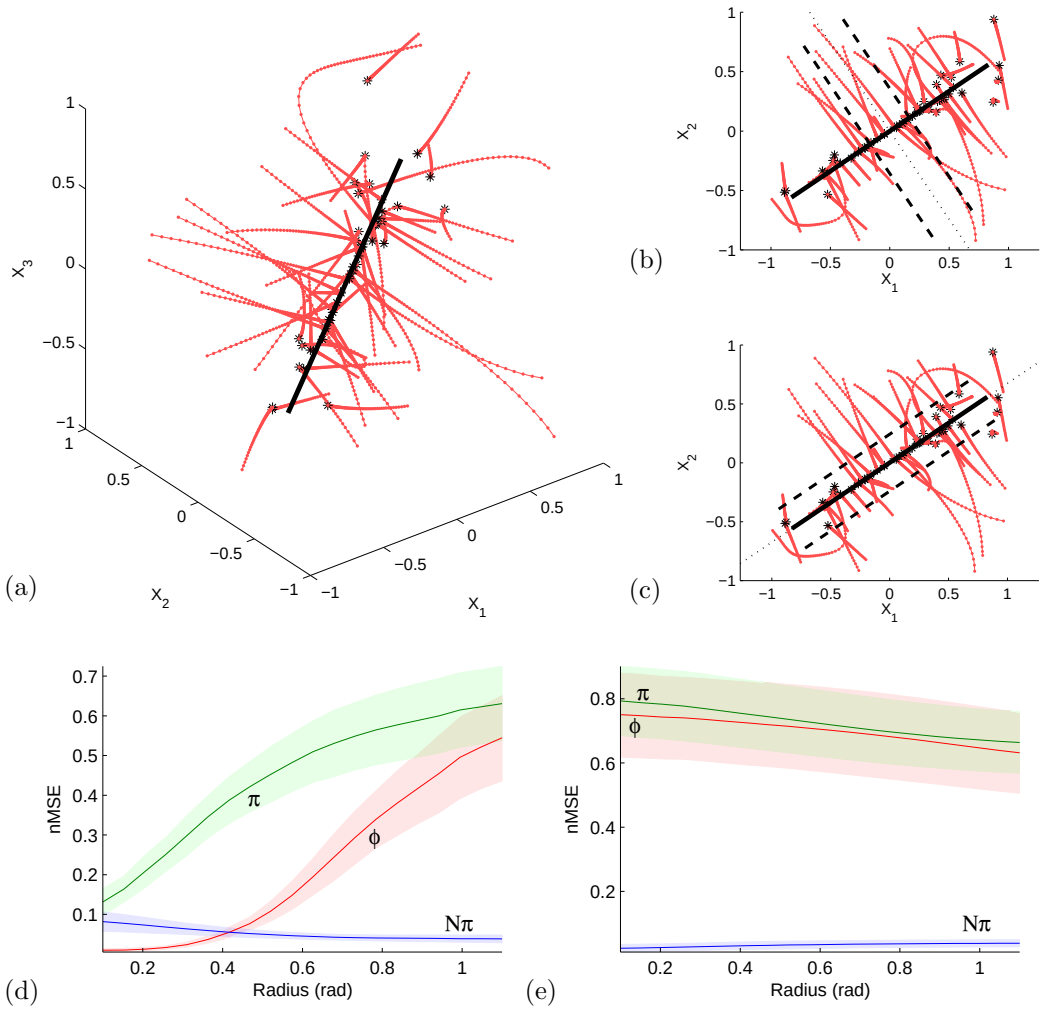


Figure 9. (a) Trajectories in state-space for the three link arm subject to random planar constraints on the hand. (b) and (c) show projections onto the first two joints of the arm, and also indicate the line attractor (solid black line). We sampled the nMSE at increasing distances along the line (b) and radially outward from it (c). Plots (d) and (e) depict the cumulative nMSE of the potential ϕ, policy $\boldsymbol{\pi}$, and constrained policy ($\mathbf{N}\boldsymbol{\pi}$) as a function of the distance measures from (b) and (c), respectively.

to complex constraints in high-dimensional movement systems. The core ingredient is an algorithm for aligning local models of the potential, which leads to a convex optimisation problem.

Under the analytical limitations of what can be learnt in this setting, our method performs remarkably well: Ultimately, the ability to learn the nullspace potential depends on the constraints. Given a pathological set of constraints, one can never hope to recover the potential. However, using our method, motion data under different constraints can be combined to learn a potential that is consistent with the observations. With a reasonably rich set of constraints, we can recover the unconstrained policy with high accuracy, and we can generalise to predict behaviour under different constraints.

For future research, we plan to work on a more principled selection of the smoothing parameter σ^2, which probably will include varying that parameter across the state space. Another possibility is to align the trajectories by hierarchical grouping, fitting more complex models to the growing groups.

Appendix A. Error Measures

In order to measure the performance of our algorithm we define the following two metrics. Firstly, the normalised *unconstrained policy error* (nUPE) is

$$E_{upe}[\tilde{\boldsymbol{\pi}}] = \frac{1}{N\sigma_{\boldsymbol{\pi}}^2} \sum_{n=1}^{N} ||\boldsymbol{\pi}(\mathbf{x}_n) - \tilde{\boldsymbol{\pi}}(\mathbf{x}_n)||^2 \tag{A1}$$

where N is the number of data points, $\boldsymbol{\pi}(\mathbf{x}_n)$ and $\tilde{\boldsymbol{\pi}}(\mathbf{x}_n)$ are the (unconstrained) true and learnt policy predictions at the points $\mathbf{x}_n$ and $\sigma_{\boldsymbol{\pi}}^2$ is the variance in the true policy over those points. The nUPE measures the difference between samples of the (unconstrained) true and learnt policies, normalised by the variance. Since the primary goal of our algorithm is to find a good approximation of the unconstrained policy, a low nUPE indicates good performance. Note also that the nUPE also gives an indication of how well the learnt policy will generalise over different constraints, since if the learnt policy closely matches the true unconstrained policy, then it will also closely match the true policy under any arbitrary projection (constraint).

The second measure we define is the normalised *constrained policy error* (nCPE)

$$E_{cpe}[\tilde{\boldsymbol{\pi}}] = \frac{1}{N\sigma_{\boldsymbol{\pi}}^2} \sum_{n=1}^{N} ||\mathbf{N}_n \left(\boldsymbol{\pi}(\mathbf{x}_n) - \tilde{\boldsymbol{\pi}}(\mathbf{x}_n)\right)||^2 \tag{A2}$$

where $\mathbf{N}_n$ denotes the constraint (projection) matrix for the n-th point. The nCPE measures the difference between the true and learnt policies under the projections $\mathbf{N}_n$. The significance of the nCPE is that it allows one to measure the accuracy of the learnt policy under a specific set of constraints (i.e. those encoded by the projections $\mathbf{N}_n$). For example, if we chose $\mathbf{N}_n$ as the set of projections corresponding to the constraints in force in the training data, then we can assess how well our model will perform under those same constraints. Alternatively, if we chose $\mathbf{N}_n$ corresponding to a set of novel, unseen constraints we we can directly measure how well the policy generalises to predict under those new constraints.

While not directly of interest in terms of controller performance, the normalised error in the learnt potential can provide information about the algorithm as a whole. It is given by

$$E_{pot}[f] = \frac{1}{N\sigma_{\phi}^2} \sum_{n=1}^{N} \big(f(\mathbf{x}_n) - \phi(\mathbf{x}_n) - \mu\big)^2 \,, \quad \mu = \frac{1}{N} \sum_{n=1}^{N} \big(f(\mathbf{x}_n) - \phi(\mathbf{x}_n)\big), \tag{A3}$$

where σ_{ϕ}^2 denotes the variance of the true potential. Please note that we subtract the mean difference μ between the two quantities to remove the irrelevant global offset of the potentials. While the potential error also depends on the accuracy of the regression method (LWPR), it is mostly determined by which offsets b_k we pick for the training trajectories. We can measure this part by the *normalised alignment error*

$$E_{align}[\mathbf{b}] = \frac{1}{N\sigma_{\phi}^2} \sum_{n=1}^{N} \big(\hat{\phi}(\mathbf{x}_n) - \phi(\mathbf{x}_n) - \nu\big)^2 \,, \quad \nu = \frac{1}{N} \sum_{n=1}^{N} \big(\hat{\phi}(\mathbf{x}_n) - \phi(\mathbf{x}_n)\big), \tag{A4}$$

where the notation $\hat{\phi}(\mathbf{x}_n)$ is understood to already include the proper offset, that is, $\hat{\phi}(\mathbf{x}_n) = \hat{\phi}_{kn'} + b_k$ if the test point $\mathbf{x}_n$ was held out from the k-th trajectory we

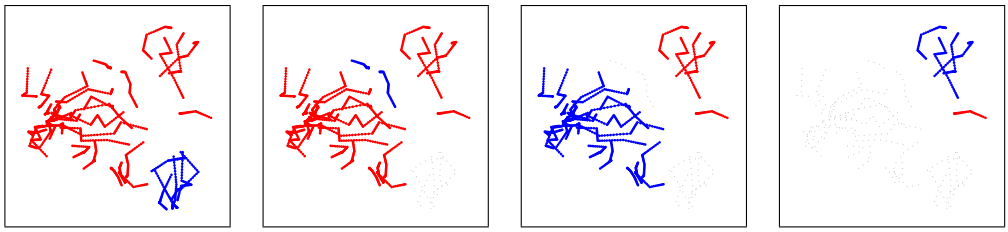


Figure B1. Illustration of our recursive outlier detection scheme. At any stage, we look for non-trivial small eigenvalues of the alignment Hessian, and if those exist, we split the trajectories into 2 independent groups (red and blue). From left to right: 1) top-level partitioning 2) splitting up the red group from step 1, 3) splitting the red group from step 2, 4) splitting the red group from step 3. The largest connected group consists of the blue trajectories from step 3, which we use for training the global model.

trained the model on.

Appendix B. Recursive Bi-partitioning for Outlier Detection

In the following, we describe our mechanism for detecting trajectories (or groups thereof) that we need to discard from the training set before learning a global model of the potential. To this end, similarly to spectral clustering, we look at the eigenvectors belonging to all small eigenvalues of the Hessian $\mathbf{H}$ (16). Let

$$\mathbf{V} = (\mathbf{v}_1 \mathbf{v}_2 \dots \mathbf{v}_n)^T \quad \text{where} \quad \lambda_i \mathbf{v}_i = \mathbf{H}\mathbf{v}_i \ , \quad \lambda_i < 10^{-8}. \tag{B1}$$

That is, if $\mathbf{H}$ was calculated from 100 trajectories and has $n = 7$ small eigenvalues, $\mathbf{V}$ would be a 7×100 matrix. We then cluster the columns of $\mathbf{V}$ into two centres $\mathbf{c}_1, \mathbf{c}_2 \in \mathbb{R}^n$. Since each column of $\mathbf{V}$ represents a trajectory, we effectively partition the training data into two groups whose relative potential offset has negligible influence on the alignment objective function (16). For both groups, we repeat the process using corresponding sub-matrices of $\mathbf{H}$. That is, we recursively split up our trajectories into groups until there is only one zero eigenvalue left in each group (corresponding to $\mathbf{v} = \mathbf{1}$, the constant shift of all trajectories in that group). The process is visualised in Fig. B1.

References

Alissandrakis A, Nehaniv C, Dautenhahn K. 2007. Correspondence Mapping Induced State and Action Metrics for Robotic Imitation. IEEE Transactions on Systems, Man and Cybernetics 37(2):299–307.

Antonelli G, Arrichiello F, Chiaverini S. 2005. The Null-space-based Behavioral Control for Soccer-playing Mobile RobotsIn: *IEEE Int. Conf. Advanced Intelligent Mechatronics*.

Atkeson C, Schaal S. 1997. Robot Learning from DemonstrationIn: *Int. Conf. Machine Learning*.

Billard A, Calinon S, Dillmann R, Schaal S. 2007. Robot Programming by Demonstration. In: Handbook of Robotics. MIT Press.

Calinon S, Billard A. 2007. Learning of Gestures by Imitation in a Humanoid RobotIn: *Imitation & Social Learning in Robots, Humans & Animals: Behavioural, Social & Communicative Dimensions*.

Chajewska U, Koller D, Ormoneit D. 2001. Learning an Agent's Utility Function by Observing BehaviorIn: *Int. Conf. Machine Learning*.

Chajewska U, Getoor L, Norman J, Shahar Y. 1998. Utility Elicitation as a Classification ProblemIn: *Proc. Conf. Uncertainty in Artificial Intelligence*.

Chalodhorn R, Grimes DB, Maganis GY, Rao RP, Asada M. 2006. Learning Humanoid Motion Dynamics through Sensory-motor Mapping in Reduced Dimensional SpaceIn: *ICRA*.

Chaumette F, Marchand A. 2001. A Redundancy-based Iterative Approach for Avoiding Joint Limits: Application to Visual Servoing. IEEE Trans. Robotics and Automation 17:719–730.

Choi S, Kim B. 2000. Obstacle Avoidance Control for Redundant Manipulators Using Collidability Measure. Robotica 18:143–151.

Conner D, Rizzi A, Choset H. 2003. Composition of Local Potential Functions for Global Robot Control and NavigationIn: *IEEE Int. Conf. Intelligent Robots and Systems*.

D'Souza A, Vijayakumar S, Schaal S. 2001. Learning Inverse KinematicsIn: *IEEE Int. Conf. Intelligent Robots and Systems*.

English J, Maciejewski A. 2000. On the Implementation of Velocity Control for Kinematically Redundant Manipulators. IEEE Trans. Sys., Man and Cybernetics 30:233–237.

Gienger M, Janssen H, Goerick C. 2005. Task-oriented Whole Body Motion for Humanoid RobotsIn: *Humanoids*.

Grimes D, Chalodhorn R, Rao R. 2006. Dynamic Imitation in a Humanoid Robot through Nonparametric Probabilistic InferenceIn: *Robotics: Science and Systems*.

Grimes D, Rashid D, Rao R. 2007. Learning Nonparametric Models for Probabilistic ImitationIn: *NIPS*.

Guenter F, Hersch M, Calinon S, Billard A. 2007. Reinforcement Learning for Imitating Constrained Reaching Movements. RSJ Advanced Robotics, Special Issue on Imitative Robots 21:1521–1544.

Howard M, Gienger M, Goerick C, Vijayakumar S. 2006. Learning Utility Surfaces for Movement SelectionIn: *IEEE Int. Conf. Robotics and Biomimetics*.

Howard M, Klanke S, Gienger M, Goerick C, Vijayakumar S. 2008. Learning Potential-based Policies from Constrained MotionIn: *IEEE Int. Conf. on Humanoid Robots*.

Howard M, Vijayakumar S. 2007. Reconstructing Null-space Policies Subject to Dynamic Task Constraints in Redundant ManipulatorsIn: *W.S. Robotics and Mathematics*.

Ijspeert A, Nakanishi J, Schaal S. 2002. Movement Imitation with Nonlinear Dynamical Systems in Humanoid RobotsIn: *ICRA*.

Ijspeert A, Nakanishi J, Schaal S. 2003. Learning Attractor Landscapes for Learning Motor PrimitivesIn: *NIPS*.

Inamura T, Toshima I, Tanie H, Nakamura Y. 2004. Embodied Symbol Emergence based on Mimesis Theory. Int. J. Robotics Research 23:363–377.

Itiki C, Kalaba R, Udwadia F. 1996. Inequality Constraints in the Process of Jumping. Applied Mathematics and Computation 78:163–173.

Kannan R, Vempala S, Vetta A. 2004. On Clusterings: Good, Bad and Spectral. J. of the ACM 51:497–515.

Khatib O. 1985. Real-Time Obstacle Avoidance for Manipulators and Mobile RobotsIn: *ICRA*.

Khatib O. 1987. A Unified Approach for Motion and Force Control of Robot Manipulators: The Operational Space Formulation. IEEE J. Robotics and Automation RA-3:43–53.

Körding K, Fukunaga I, Howard I, Ingram J, Wolpert D. 2004. A Neuroeconomics Approach to Inferring Utility Functions in Sensorimotor Control. PLoS Biol. 2:330.

Körding K, Wolpert D. 2004. The Loss Function of Sensorimotor LearningIn: *Proc. National Academy of Sciences*.

Liégeois A. 1977. Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms. IEEE Trans. Sys., Man and Cybernetics 7:868–871.

Mattikalli R, Khosla P. 1992. Motion Constraints from Contact geometry: Representation and AnalysisIn: *ICRA*.

Murray R, Li Z, Sastry S. 1994. A Mathematical Introduction to Robotic Manipulation. CRC Press.

Mussa-Ivaldi F. 1997. Nonlinear Force Fields: A Distributed System of Control Primitives for Representing and Learning MovementsIn: *Proc. CIRA'97*.

Nakamura Y. 1991. Advanced Robotics: Redundancy and Optimization. Addison Wesley.

Nakanishi J, Morimoto J, Endo G, Cheng G, Schaal S, Kawato M. 2004. Learning from Demonstration and Adaptation of Biped Locomotion. Robotics and Autonomous Systems 47:79–91.

Ohta K, Svinin M, Luo Z, Hosoe S, Laboissiere R. 2004. Optimal trajectory formation of constrained human arm reaching movements. Biol. Cybern. 91:23–36.

Park J, Khatib O. 2006. Contact Consistent Control Framework for Humanoid RobotsIn: *ICRA*.

Peters J, Mistry M, Udwadia F, Nakanishi J, Schaal S. 2008. A Unifying Framework for Robot Control with Redundant DOFs. Autonomous Robots J. 24:1–12.

Peters J, Schaal S. 2008. Learning to Control in Operational Space. Int. J. Robotics Research 27:197–212.

Ren J, McIsaac KA, Patel RV. 2006. Modified Newton's Method Applied to Potential Field-based Navigation for Mobile Robots. IEEE Transactions on Robotics .

Rimon E, Koditschek D. 1992. Exact Robot Navigation Using Artificial Potential Functions. IEEE Trans. Robotics and Automation 8:501–518.

Sapio VD, Khatib O, Delp S. 2006. Task-Level Approaches for the Control of Constrained Multibody Systems.

Sapio VD, Warren J, Khatib O, Delp S. 2005. Simulating the Task-level Control of Human Motion: A Methodology and Framework for Implementation. The Visual Computer 21(5):289–302.

Schaal S, Ijspeert A, Billard A. 2003. Computational Approaches to Motor Learning by Imitation. Phil. Trans.: Biological Sciences 358:537–547.

Sentis L, Khatib O. 2004. Task-oriented Control of Humanoid Robots through PrioritizationIn: *IEEE Int. Conf. on Humanoid Robots*.

Sentis L, Khatib O. 2005. Synthesis of Whole-body Behaviors through Hierarchical Control of Behavioral Primitives. International Journal of Humanoid Robotics 2:505–518.

Sentis L, Khatib O. 2006. A Whole-body Control Framework for Humanoids Operating in Human EnvironmentsIn: *ICRA*.

Sugiura H, Gienger M, Janssen H, Goerick C. 2007. Real-time Collision Avoidance with Whole Body Motion Control for Humanoid RobotsIn: *IROS*.

Sutton R, Barto A. 1998. Reinforcement Learning: An Introduction. MIT Press.

Svinin M, Odashima T, Ohno S, Luo Z, Hosoe S. 2005. An Analysis of Reaching Movements in Manipulation of Constrained Dynamic ObjectsIn: *IROS*.

Takano W, Yamane K, Sugihara T, Yamamoto K, Nakamura Y. 2006. Primitive Communication based on Motion Recognition and Generation with Hierarchical Mimesis ModelIn: *ICRA*.

Todorov E. 2006. Optimal Control Theory. In: Bayesian Brain. MIT Press.

Verbeek J, Roweis S, Vlassis N. 2004. Non-linear CCA and PCA by alignment of local modelsIn: *NIPS*.

Verbeek J. 2006. Learning non-linear image manifolds by combining local linear models. IEEE Trans. Pattern Analysis & Machine Intelligence 28:1236–1250.

Vijayakumar S, D'Souza A, Schaal S. 2005. Incremental Online Learning in High Dimensions. Neural

Comp. 17:2602–2634.
Yoshikawa T. 1985. Manipulability of Robotic Mechanisms. Int. J. Robotics Research 4:3–9.