Preprints of the 21st IFAC World Congress (Virtual)
Berlin, Germany, July 12-17, 2020

On the synthesis of control policies from example datasets

Davide Gagliardi [*] **Giovanni Russo** [**]

[*] *School of Electrical and Electronic Engineering, University College Dublin, Ireland (e-mail: davide.gagliardi@ucd.ie).*
[**] *Department of Information and Electrical Engineering and Applied Mathematics, University of Salerno (e-mail: giovarusso@unisa.it).*

Keywords: Learning for control, Inverse Learning, Control from demonstrations

1. EXTENDED ABSTRACT

A framework that is becoming particularly appealing to design control algorithms is that of devising the control policy from examples (or demonstrations), see e.g. Hanawal et al. (2019); Wabersich and Zeilinger (2018) and references therein. At their roots these *control from demonstration techniques*, which are gaining considerable attention under the label of Inverse Reinforcement Learning (IRL), rely on Inverse Optimal Control and Optimization Bryson (1996). Today, IRL/control is recognized as an appealing framework to learn policies from *success stories* Argall et al. (2009) and potential applications include planning Englert et al. (2017) and preferences/prescriptions learning Xu and Paschalidis (2019).

There is then no surprise that, over the years, a number of techniques have been developed to address the problem of devising control policies from demonstrations, mainly in the context of Markov Decision Processes (MDPs). Results include Ratliff et al. (2009), which leverages a linear programming approach, Ziebart et al. (2008) that uses the maximum entropy principle and Ramachandran and Amir (2007) that leverages Bayesian statistics.

In this context, the main contributions of this extended abstract can be summarized as follows. First, we introduce an approach to synthesize control policies from examples which is based on the Fully Probabilistic Design (FPD) Kárný (1996); Kárný and Guy (2006); Herzallah (2015); Pegueroles and Russo (2019); Krn and Kroupa (2012). This approach formalizes the control problem as an optimization problem where the Kullback-Leibler Divergence (see Section 1.2) between an *ideal* probability density function (pdf, obtained from e.g. demonstrations) and the pdf modeling the system/plant is minimized. The main technical novelty of our results with respect to the classic works on FPD lies in the fact that we explicitly embed actuation constraints in our formulation, thus solving an optimization problem where the Kullback-Leibler Divergence is minimized subject to constraints on the control variable. By relying on the FPD, one of the main advantages of our results over classic IRL/Control approaches is that policies can be synthesized from data without requiring linearity of the underlying system. Moreover, by embedding actuation constraints into the problem formulation and by solving the resulting optimization, we can export the policy that has been learned on other systems that have different actuation capabilities. As an additional contribution, we devise from our theoretical results an algorithmic procedure. The key reference applications over which the algorithm was tested involved an autonomous driving use case.

[⋆] The authors were supported by Science Foundation Ireland (SFI) under Grant Number 16/RC/3872

1.1 Notation

Sets, as well as operators, are denoted with *calligraphic* characters, while vector quantities are denoted in **bold**. Let n_z be a positive integer and consider the measurable space $(\mathcal{Z}, \mathcal{F}_z)$, with $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$ and with $\mathcal{F}_z$ being a σ-algebra on $\mathcal{Z}$. Then, the random variable on $(\mathcal{Z}, \mathcal{F}_z)$ is denoted by $\mathbf{Z}$ and its realization is denoted by $\mathbf{z}$. The pdf of $\mathbf{Z}$ is denoted by $f(\mathbf{z})$ (or, equivalently, by $f_{\mathbf{Z}}$) and its support is denoted by $\mathrm{S}(f)$. We recall that, given a function $\mathbf{h}(\cdot)$, the expectation of $\mathbf{h}(\mathbf{z})$, i.e. $\mathbb{E}_f[\mathbf{h}(\mathbf{Z})]$ is defined as $\mathbb{E}_f[\mathbf{h}(\mathbf{Z})] := \int_{\mathrm{S}(f)} \mathbf{h}(\mathbf{z}) f(\mathbf{z}) d\mathbf{z}$. For notational convenience, whenever it is clear from the context, we omit the domain of integration as well as the subscript in the expectation. The conditional probability density function (cpdf) of $\mathbf{Z}$ with respect to the random variable $\mathbf{Y}$ is denoted by $f(\mathbf{z}|\mathbf{y})$ and sometimes we will use the shorthand notation $\tilde{f}_{\mathbf{Z}}$. Given $\mathcal{Z} \subseteq \mathbb{R}^{n_z}$, its indicator function is denoted by $\mathbb{1}_{\mathcal{Z}}(\mathbf{z})$: $\mathbb{1}_{\mathcal{Z}}(\mathbf{z}) = 1$, $\forall \mathbf{z} \in \mathcal{Z}$ and 0 otherwise. Finally, we will also make use of the internal product between tensors, which is denoted by $\langle \cdot, \cdot \rangle$.

1.2 The Kullback-Leibler divergence

The control problem considered in this abstract will be stated (see Section 1.4) in terms of the Kullback-Leibler (KL, Kullback and Leibler (1951)) divergence:

Definition 1. (Kullback-Leibler(KL) divergence). Consider two pdfs, $f_1(\mathbf{z})$ and $f_2(\mathbf{z})$, with $f_1(\mathbf{z})$ being absolutely continuous with respect to $f_2(\mathbf{z})$. Then, the KL-divergence of $f_1(\mathbf{z})$ with respect to $f_2(\mathbf{z})$ is

$$\mathcal{D}_{\mathrm{KL}}(f_1 || f_2) := \int_{\mathrm{S}(f_1)} f_1 \ln\left(\frac{f_1}{f_2}\right) d\mathbf{z}. \qquad (1)$$

Intuitively, $\mathcal{D}_{\mathrm{KL}}(f_1||f_2)$ measures how well $f_1(\mathbf{z})$ approximates $f_2(\mathbf{z})$. We also recall that $\mathcal{D}_{\mathrm{KL}}(f_1||f_2)$ exists if $\mathrm{S}(f_1) \subseteq \mathrm{S}(f_2)$. We assume that the KL-divergence for the pdfs of our interest exists.

1.3 System description

Let: (i) $\mathcal{K} := \{k\}_{k=1}^{n}$, $\mathcal{K}_0 := \mathcal{K} \cup \{0\}$ and $\mathcal{T} := \{t_k : k \in \mathcal{K}_0\}$ be the time horizon over which the system is observed; (ii) $\mathbf{x}_k \in \mathbb{R}^{d_x}$ and $\mathbf{u}_k \in \mathbb{R}^{d_u}$ be, respectively, the system state and input at time $t_k \in \mathcal{T}$; (ii) $\mathbf{d}_k := (\mathbf{x}_k, \mathbf{u}_k)$ be the data collected from the system at time $t_k \in \mathcal{T}$ and $\mathbf{d}^k$ the data collected from $t_0 \in \mathcal{T}$ up to time $t_k \in \mathcal{T}$ $(t_k > t_0)$. Then, as shown in e.g. Peterka (1981), the system behavior can be described via the joint pdf of the observed data, say $f(\mathbf{d}^n)$. Then, as shown in the same paper, the application of the chain rule for probability density functions leads to the following factorization for $f(\mathbf{d}^n)$:

$$f(\mathbf{d}^n) = \prod_{k \in \mathcal{K}} f(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1}) f(\mathbf{u}_k|\mathbf{x}_{k-1}) f(\mathbf{x}_0). \qquad (2)$$

Throughout this abstract we will refer to (2) as the *probabilistic description of the closed loop system*, or simply say that (2) is our *closed loop* system.

Remark 1. The cpdf $f(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1})$ describes the system behavior at time t_k, given the previous state and the input at time t_k. In turn, the input is also generated from the cdpf of a *randomized control algorithm* $f(\mathbf{u}_k|\mathbf{x}_{k-1})$, which indeed returns the input given the previous system state. We also note that initial conditions are embedded in the probabilistic system description through the prior $f(\mathbf{x}_0)$.

In the following we will use the *shorthand* notations $\tilde{f}^k_{\mathbf{X}} := f(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1})$, $\tilde{f}^k_{\mathbf{U}} := f(\mathbf{u}_k|\mathbf{x}_{k-1})$, $f_0 := f(\mathbf{x}_0)$ and $f^n := f(\mathbf{d}^n)$ so that (2) can be written in the more compact form

$$f^n = \prod_{k \in \mathcal{K}} \tilde{f}^k_{\mathbf{X}} \tilde{f}^k_{\mathbf{U}} f_0 = \tilde{f}^n f_0, \quad \tilde{f}^n := \prod_{k \in \mathcal{K}} \tilde{f}^k_{\mathbf{X}} \tilde{f}^k_{\mathbf{U}}. \qquad (3)$$

1.4 The control problem

Our goal is to synthesize the control pdf $f(\mathbf{u}_k|\mathbf{x}_{k-1})$ so that the behavior illustrated via an example dataset, say $\mathbf{d}^n$, can be tracked by system (3) subject to its actuation constraints. As in Kárný (1996); Quinn et al. (2016); Pegueroles and Russo (2019); Kárný and Guy (2006); Herzallah (2015) the behavior illustrated in the example dataset can be specified through the reference pdf $g(\mathbf{d}^n)$ extracted from the dataset (as e.g. its empirical distribution). Following the chain rule for pdfs we have $g(\mathbf{d}^n) := \prod_{k \in \mathcal{K}} g(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1}) g(\mathbf{u}_k|\mathbf{x}_{k-1}) g(\mathbf{x}_0)$. Again, by setting $\tilde{g}^k_{\mathbf{X}} := g(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1})$, $\tilde{g}^k_{\mathbf{U}} := g(\mathbf{u}_k|\mathbf{x}_{k-1})$, $g_0 := g(\mathbf{x}_0)$ and $g^n := g(\mathbf{d}^n)$ we get:

$$g^n = \prod_{k \in \mathcal{K}} \tilde{g}^k_{\mathbf{X}} \tilde{g}^k_{\mathbf{U}} g_0 = \tilde{g}^n g_0, \qquad (4)$$

where $\tilde{g}^n := \prod_{k \in \mathcal{K}} \tilde{g}^k_{\mathbf{X}} \tilde{g}^k_{\mathbf{U}}$.

Our tracking problem can then be recast as the problem of designing $f(\mathbf{u}_k|\mathbf{x}_{k-1})$ so that f^n approximates g^n. This leads to the following formalization:

Problem 1. Determine the sequence of cpdfs, say $\left\{\left(\tilde{f}^k_{\mathbf{U}}\right)^*\right\}_{k \in \mathcal{K}}$, solving the nonlinear program

$$\begin{aligned} \min_{\{\tilde{f}^k_{\mathbf{U}}\}_{k \in \mathcal{K}}} \quad & \mathcal{D}_{\mathrm{KL}}(f^n || g^n) \\ \text{s.t.} \quad & \mathbb{E}_{\tilde{f}^k_{\mathbf{U}}}\left[\tilde{\mathbf{h}}_{\mathbf{u},k}(\mathbf{U})\right] = \tilde{\mathbf{H}}_{\mathbf{u},k}, \quad k \in \mathcal{K} \end{aligned} \qquad (5)$$

We note that the program constraints can be equivalently written as $\int_{\mathrm{S}(\tilde{f}^k_{\mathbf{U}})} \tilde{f}^k_{\mathbf{U}} \; \tilde{\mathbf{h}}_{\mathbf{u},k}(\mathbf{u}_k) \; d\mathbf{u}_k = \tilde{\mathbf{H}}_{\mathbf{u},k},$. Finally, the constraints of the program are time-varying and the number of constraints can change over time (the number of constraints at time t_k is denoted by $c_{\mathbf{u},k}$). Indeed, in the constraints of (5): (i) $\tilde{\mathbf{H}}_{\mathbf{u},k}$ is a (column) vector of coefficients, i.e. $\tilde{\mathbf{H}}_{\mathbf{u},k} := \left[H_{\mathbf{u},0,k}, \mathbf{H}^T_{\mathbf{u},k}\right]^T$ and $\tilde{\mathbf{h}}_{\mathbf{u},k}(\mathbf{z}) := \left[h_{\mathbf{u},0,k}, \mathbf{h}^T_{\mathbf{u},k}\right]^T(\mathbf{z})$; (ii) $\mathbf{H}_{\mathbf{u},k} \in \mathbb{R}^{c_{\mathbf{u},k}}$ and $\mathbf{h}_{\mathbf{u},k} : \mathrm{S}(\tilde{f}^k_{\mathbf{U}}) \mapsto \mathbb{R}^{c_{\mathbf{u},k}}$; (iii) $H_{\mathbf{u},0,k} := 1$ and $h_{\mathbf{u},0,k}(\mathbf{z}) := \mathbb{1}_{\mathcal{U}_k}(\mathbf{z})$ ensure that the solution of the program is a cpdf.

1.5 Technical Results

For the sake of completeness, we now report the main technical results behind the algorithm of Section 1.6. The proofs will be presented elsewhere.

Lemma 1. Let $\mathbf{Z}$ be a random variable on the measurable space $(\mathcal{Z}, \mathcal{F}_z)$, $f := f_{\mathbf{Z}}(\mathbf{z})$, $g := g_{\mathbf{Z}}(\mathbf{z})$ be two probability distributions over $(\mathcal{Z}, \mathcal{F}_z)$, $\alpha : \mathcal{Z} \mapsto \mathbb{R}^+_0$ be a *nonnegative* function of $\mathbf{Z}$, *integrable* under the measure given by $f_{\mathbf{Z}}(\mathbf{z})$. Given this set-up, assume that $f_{\mathbf{Z}}(\mathbf{z})$ satisfies the following set of algebraically independent constraints

$$\int f_{\mathbf{Z}}(\mathbf{z}) \; \tilde{\mathbf{h}}(\mathbf{z}) \; d\mathbf{z} = \tilde{\mathbf{H}}, \qquad (6)$$

where: (i) $\tilde{\mathbf{h}}(\mathbf{z}) := \left[h_0, \mathbf{h}^T\right]^T(\mathbf{z})$, with $h_0(\mathbf{z}) := \mathbb{1}_{\mathcal{S}(\mathbf{Z})}(\mathbf{z})$ and $\mathbf{h} : \mathcal{Z} \mapsto \mathbb{R}^{c_{\mathbf{z}}}$ being a measurable map; (ii) $\tilde{\mathbf{H}}(\mathbf{z}) := \left[H_0, \mathbf{H}^T\right]^T$ with $H_0 := 1$ and $\mathbf{H} \in \mathbb{R}^{c_{\mathbf{z}}}$ being a vector of constants. Then, the solution of the constrained optimization problem

$$\begin{aligned} \min_{f_{\mathbf{Z}}} \quad & \mathcal{D}_{\mathrm{KL}}(f||g) + \int f_{\mathbf{Z}}(\mathbf{z}) \, \alpha(\mathbf{z}) \; d\mathbf{z} \\ \text{s.t.} \quad & \text{constraints in (6)} \end{aligned} \qquad (7)$$

is the pdf

$$f^*_{\mathbf{Z}}(\mathbf{z}) = g(\mathbf{z}) \frac{e^{-\{\alpha(\mathbf{z}) + \langle \boldsymbol{\lambda}^*, \mathbf{h}(\mathbf{z}) \rangle\}}}{e^{1+\lambda^*_0}}, \qquad (8)$$

where λ^*_0 and $\boldsymbol{\lambda}^* = [\lambda^*_1, \ldots, \lambda^*_{c_{\mathbf{z}}}]^T$ are the *Lagrange multipliers* associated to the constraints. Moreover, the corresponding minimum of the cost function $\mathcal{J}(f) := \mathcal{D}_{\mathrm{KL}}(f||g) + \int f_{\mathbf{Z}}(\mathbf{z}) \, \alpha(\mathbf{z}) \; d\mathbf{z}$ is

$$\mathcal{J}^* := \mathcal{J}(f^*) = -\left(1 + \lambda^*_0 + \langle \boldsymbol{\lambda}^*, \mathbf{H} \rangle\right). \qquad (9)$$

Note that, in Lemma 1, the optimal solution $f^*_{\mathbf{Z}}(\mathbf{z})$ depends on the Lagrange multipliers (LMs) λ^*_0 and $\boldsymbol{\lambda}^*$. While λ^*_0 can be obtained by integration, all the other LMs need to be computed numerically. With the next result, we propose a strategy for finding the LMs $\boldsymbol{\lambda}^*$. In particular, our key idea is to recast the problem of finding the solutions of nonlinear equations as a minimization problem. In general, the approach can be also used to fit the parameters of a pdf so that it meets a set of pre-specified constrains (for example, to find pdfs that satisfy the Maximum Entropy principle).

Lemma 2. Let: (i) $\mathcal{Z} \subseteq \mathbb{R}^{n_{\mathbf{z}}}$ and $\tilde{\boldsymbol{\Theta}} \subseteq \mathbb{R}^{n_{\mathbf{z}}}$; (ii) $\hat{f}_1 : \mathcal{Z} \mapsto \hat{f}_1(\mathbf{z})$ be a positive and integrable function on $\mathcal{Z}$; (iii) $\hat{f}_2 : (\mathcal{Z} \times \tilde{\boldsymbol{\Theta}}) \mapsto \hat{f}_1(\mathbf{z}) e^{-\langle \tilde{\boldsymbol{\theta}}, \tilde{\mathbf{h}}(\mathbf{z}) \rangle}$, where $\tilde{\mathbf{h}} = \left[\tilde{\mathbf{h}}_1(\mathbf{z}), \ldots, \tilde{\mathbf{h}}_{c_{\mathbf{z}}}(\mathbf{z})\right]^T : \mathcal{Z} \mapsto \mathbb{R}^{c_{\mathbf{z}}}$. Consider the set of algebraically independent equations

$$\int_{\mathcal{Z}} \hat{f}_2\left(\mathbf{z}, \tilde{\boldsymbol{\theta}}\right) \tilde{\mathbf{h}}_i(\mathbf{z}) \; d\mathbf{z} = \tilde{\mathbf{H}}_i, \quad i = 1, \ldots, c_{\mathbf{z}}, \qquad (10)$$

where $\tilde{\mathbf{H}} := \left[\tilde{\mathbf{H}}_1, \ldots, \tilde{\mathbf{H}}_{c_{\mathbf{z}}}\right]^T \in \mathbb{R}^{c_{\mathbf{z}}}$. Then, the unique solution, say $\tilde{\boldsymbol{\theta}}^*$, of the minimization problem

$$\min_{\tilde{\boldsymbol{\theta}}} \mathcal{J}\left(\tilde{\boldsymbol{\theta}}\right), \tag{11}$$

with $\mathcal{J}\left(\tilde{\boldsymbol{\theta}}\right) := \langle\tilde{\boldsymbol{\theta}}, \tilde{\mathbf{H}}\rangle + \int_{\mathcal{Z}} \hat{f}_2\left(\mathbf{z}, \tilde{\boldsymbol{\theta}}\right) d\mathbf{z}$ is also a solution of (10).

The main result behind the algorithm of Section 1.6, the proof of which leverages the above technical lemmas, is presented next.

Theorem 1. The solution, $\left(\tilde{f}^k_{\mathbf{U}}\right)^* = f^*\left(\mathbf{u}_k|\mathbf{x}_{k-1}\right)$, of the control Problem 1 is

$$\left(\tilde{f}^k_{\mathbf{U}}\right)^* = \tilde{g}^k_{\mathbf{U}} \frac{e^{-\{\hat{\omega}(\mathbf{u}_k, \mathbf{x}_{k-1}) + \langle \boldsymbol{\lambda}^*_{\mathbf{u},k}, \mathbf{h}_{\mathbf{u},k}(\mathbf{u}_k)\rangle\}}}{e^{1+\lambda^*_{\mathbf{u},0,k}}}, \tag{12}$$

where:

(1) $\hat{\omega}(\cdot,\cdot)$ is generated via backward recursion. In particular,

$$\hat{\omega}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) = \hat{\alpha}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) + \hat{\beta}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) \tag{13}$$

and

$$\begin{aligned} \hat{\alpha}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) &:= \mathcal{D}_{\mathrm{KL}}\left(\tilde{f}^k_{\mathbf{X}} || \tilde{g}^k_{\mathbf{X}}\right) \\ \hat{\beta}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) &:= -\mathbb{E}_{\tilde{f}^k_{\mathbf{X}}}\left[\ln \hat{\gamma}\left(\mathbf{X}_k\right)\right] \end{aligned} \tag{14}$$

with terminal conditions $\hat{\beta}\left(\mathbf{u}_n, \mathbf{x}_{n-1}\right) = 0$ and $\hat{\alpha}\left(\mathbf{u}_n, \mathbf{x}_{n-1}\right) = \mathcal{D}_{\mathrm{KL}}\left(\tilde{f}^n_{\mathbf{X}} || \tilde{g}^n_{\mathbf{X}}\right)$;

(2) $\hat{\gamma}(\cdot)$ is defined as

$$\ln \hat{\gamma}\left(\mathbf{x}_{k-1}\right) := \left[\sum_{i=0}^{c_{\mathbf{u}}} \ln\left(\hat{\gamma}_{\mathbf{u},i,k}\left(\mathbf{x}_{k-1}\right)\right)\right] \tag{15}$$

and $\hat{\gamma}_{\mathbf{u},i,k}(\cdot)$ are given by

$$\hat{\gamma}_{\mathbf{u},0,k}\left(\mathbf{x}_{k-1}\right) = \exp\left\{\lambda^*_{\mathbf{u},0,k} + 1\right\} \tag{16}$$

and

$$\hat{\gamma}_{\mathbf{u},i,k}\left(\mathbf{x}_{k-1}\right) := \exp\left\{\lambda^*_{\mathbf{u},i,k}\mathbf{H}_{\mathbf{u},i,k}\right\} \quad i = 1, \ldots, c_{\mathbf{u}}. \tag{17}$$

with $\hat{\gamma}_{\mathbf{u},i,n+1}\left(\mathbf{x}_n\right) = 1 \ \forall i = 0, \ldots, c_{\mathbf{u}}$;

(3) $\lambda^*_{\mathbf{u},0,k}$ and $\boldsymbol{\lambda}^*_{\mathbf{u},k} = \left[\lambda^*_{\mathbf{u},1,k}, \ldots, \lambda^*_{\mathbf{u},c_{\mathbf{u}},k}\right]$ are the Lagrange multipliers associated to the constraints at time t_k. In particular,

$$\lambda^*_{\mathbf{u},0,k} = \\ = \ln\left\{\int \tilde{g}^k_{\mathbf{U}}\left(e^{-\{\hat{\omega}(\mathbf{u}_k, \mathbf{x}_{k-1}) + \langle\boldsymbol{\lambda}^*_{\mathbf{u},k}, \mathbf{h}_{\mathbf{u},k}(\mathbf{u}_k)\rangle\}}\right) d\mathbf{u}_k\right\} - 1,$$

while all the other LMs can be obtained numerically (via e.g. Lemma 2).

Moreover, the corresponding minimum is given by:

$$B^*_k := -\mathbb{E}_{p^{k-1}_{\mathbf{X}}}\left[\ln \hat{\gamma}\left(\mathbf{X}_{k-1}\right)\right] \tag{18}$$

where $p^k_{\mathbf{X}}$ denotes the pdf of the state at time t_k (i.e. $p^k_{\mathbf{X}} := f\left(\mathbf{x}_k\right)$).

We are now ready to introduce our control algorithm.

1.6 The algorithm

We developed an algorithmic procedure that, by leveraging the technical results introduced above, outputs the solution $\left\{\left(\tilde{f}^k_{\mathbf{U}}\right)^*\right\}_{k\in\mathcal{K}}$ to Problem 1.

Algorithm 1 Pseudo-code

```
Inputs:
$g\left(\mathbf{d}^n\right)$ and $\tilde{f}^k_{\mathbf{X}}$'s
Constraints of Problem 1 (Optional)
Output:
$\left\{\left(\tilde{f}^k_{\mathbf{U}}\right)^*\right\}_{k\in\mathcal{K}}$ solving Problem 1
Initialize
$\hat{\gamma}_{\mathbf{u},i,n+1}\left(\mathbf{x}_n\right) = 1, \forall i$ (i.e. $\hat{\beta}\left(\mathbf{u}_n, \mathbf{x}_{n-1}\right) = 0$) ;
$\hat{\gamma} \equiv \hat{\gamma}_{\mathbf{u},0,n+1}$;
for $k = n$ to 1 do
    $\hat{\alpha}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) \leftarrow \int f\left(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1}\right) \frac{f(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1})}{g(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1})} d\mathbf{x}_k$
    $\hat{\beta}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) \leftarrow \int f\left(\mathbf{x}_k|\mathbf{u}_k, \mathbf{x}_{k-1}\right)\{-\ln\left(\hat{\gamma}\left(\mathbf{x}_k\right)\right)\}$
    $\hat{\omega}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) \leftarrow \hat{\alpha}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) + \hat{\beta}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right)$
    $\hat{n}_u\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) \leftarrow g\left(\mathbf{u}_k|\mathbf{x}_{k-1}\right) \exp\left\{-\hat{\omega}\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right)\right\}$
    $\hat{\gamma}_0\left(\mathbf{x}_{k-1}\right) \leftarrow \int \hat{n}_u\left(\mathbf{u}_k, \mathbf{x}_{k-1}\right) d\mathbf{u}_k$

    $f\left(\mathbf{u}_k|\mathbf{x}_{k-1}\right) \leftarrow \frac{\hat{n}_u(\mathbf{u}_k, \mathbf{x}_{k-1})}{\hat{\gamma}_0(\mathbf{x}_{k-1})}$
    Use Lemma 2 with $\mathcal{Z} := \mathcal{S}(f\left(\mathbf{u}_k|\mathbf{x}_{k-1}\right))$, $\hat{f}_1 = f$,
$\tilde{\mathbf{H}} := \tilde{\mathbf{H}}_{\mathbf{u},k}$, $\tilde{\mathbf{h}} := \tilde{\mathbf{h}}_{\mathbf{x},k}$, $\lambda_0 := \lambda_{\mathbf{u},0,k}$, $\boldsymbol{\lambda} := \boldsymbol{\lambda}_{\mathbf{u},k}$,
$\tilde{\boldsymbol{\theta}} := \left[\theta_0, \boldsymbol{\theta}^T\right]^T = \left[1 + \lambda_0, \boldsymbol{\lambda}^T\right]^T$ to find the Lagrange
multipliers:
    $\boldsymbol{\lambda}^*_{\mathbf{u},k} = \boldsymbol{\lambda}^* \leftarrow \boldsymbol{\theta}^*$
    $\lambda^*_{\mathbf{u},0,k}\left(\mathbf{x}_{k-1}\right) = \lambda^*_0 \leftarrow \theta^*_0 - 1$
    Compute the policy and prepare variables for the
next iteration, $k-1$:
    $\left(\tilde{f}^k_{\mathbf{U}}\right)^* \leftarrow \frac{f(\mathbf{u}_k|\mathbf{x}_{k-1}) e^{-\langle\boldsymbol{\lambda}^*_{\mathbf{u},k}, \mathbf{h}_{\mathbf{u},k}(\mathbf{u}_k)\rangle}}{e^{1+\lambda^*_{\mathbf{u},0,k}}}$
    $\hat{\gamma}_{\mathbf{u},i,k}\left(\mathbf{x}_{k-1}\right) \leftarrow \exp\left\{\lambda^*_{\mathbf{u},i,k}\mathbf{H}_{\mathbf{u},i,k}\right\} \quad i = 1, \ldots, c_{\mathbf{u}}$
    $\hat{\gamma}_{\mathbf{u},0,k} = \exp\left\{\theta^*_0\right\} \leftarrow \exp\left\{\lambda^*_{\mathbf{u},0,k} + 1\right\}$
    $\hat{\gamma}\left(\mathbf{x}_{k-1}\right) \leftarrow \exp\left[\sum_{i=0}^{c_{\mathbf{u}}} \ln\left(\hat{\gamma}_{\mathbf{u},i,k}\left(\mathbf{x}_{k-1}\right)\right)\right]$
end for
```

1.7 Numerical results

We used Algorithm 1 to synthesize a control policy (from real data) that would allow an autonomous car to merge on a highway. The scenario considered in our test is schematically illustrated in Fig.1. Data were collected using the infrastructure of Griggs et al. (2019): GPS position and speed were gathered through an OBD2 connection during 100 test drives. We used the distance between the road junction point and the car position as state variable ($\mathbf{x}_k = d(t_k)$) and the car longitudinal speed as control variable ($\mathbf{u}_k = v(t_k)$). From the dataset we estimated the cpdf $\tilde{f}^k_{\mathbf{X}}$ and we used a subset of the test drives to build the example dataset. In particular, the example dataset consisted of the 20 trips that had the lowest car jerk. The example dataset was then used to estimate $\tilde{g}\left(\mathbf{d}^n\right)$ and $\tilde{g}^k_{\mathbf{U}}$ (see fig.2). Finally,

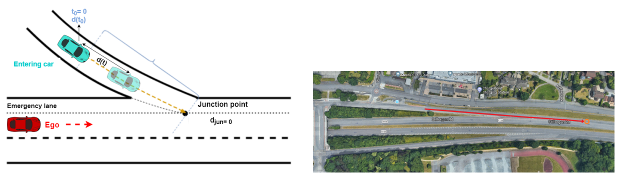

Fig. 1. Left panel: scenario illustration. Right panel: stretch of road where the experiment took place.

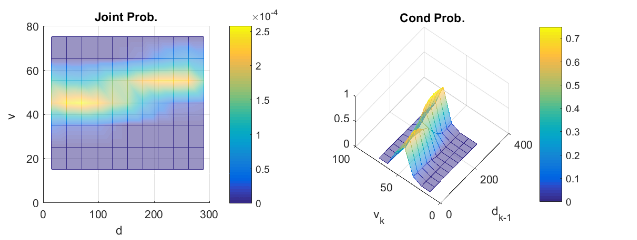

Fig. 2. Left panel: the pdf $g(\mathbf{d}^n)$ computed empirically from the dataset of 100 test drives. Right panel: $\tilde{g}^k_{\mathbf{U}}$ extracted from the example dataset.

we decided to use the following constraints in Problem 1: $\mathbf{h}_{\mathbf{u},k} = [\mathbf{u}_k, \mathbf{u}_k^2]^T$, which physically limit the maximum car speed and acceleration. The results obtained by Algorithm 1 are shown in Figure 3. The figure shows how the control policy computed by the algorithm makes the controlled closed-loop behavior *similar* to the one of the examples, while also fulfilling the constraints. In particular, the controlled pdf (in green) is flatter because we constrained it to have larger variance than the demonstrators.

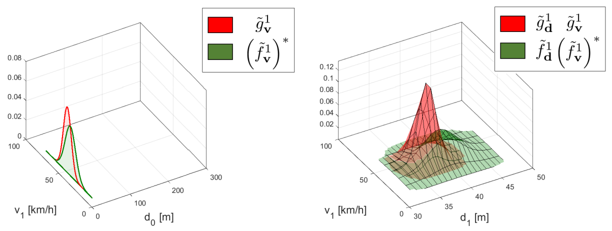

Fig. 3. Left panel: optimal policy found at $k = 1$ (green) and policy extracted from the examples (red). Right panel: closed-loop pdf resulting from the application of the optimal policy from Algorithm 1 (green) and closed-loop pdf extracted from the examples (red). The results for $k = 1$ are representative of the results for the other time instants.

1.8 Conclusions

We presented an approach to the synthesis of policies from examples. The key technical novelty is the inclusion of actuation constraints in the problem formulation. This in turn yields policies that can be exported to different systems having different actuation capabilities. After presenting the main results we introduced an algorithmic procedure and its application to an automotive use case [1].

REFERENCES

Argall, B.D., Chernova, S., Veloso, M., and Browning, B. (2009). A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5), 469 – 483. doi:https://doi.org/10.1016/j.robot.2008.10.024.

Bryson, A.E. (1996). Optimal control-1950 to 1985. *IEEE Control Systems Magazine*, 16(3), 26–33.

Englert, P., Vien, N.A., and Toussaint, M. (2017). Inverse kkt: Learning cost functions of manipulation tasks from demonstrations. *The International Journal of Robotics Research*, 36(13-14), 1474–1488. doi:10.1177/0278364917745980.

Griggs, W., Ordóñez-Hurtado, R., Russo, G., and Shorten, R. (2019). A vehicle-in-the-loop emulation platform for demonstrating intelligent transportation systems. In *Control Strategies for Advanced Driver Assistance Systems and Autonomous Driving Functions*, 133–154. Springer.

Hanawal, M., Liu, H., Zhu, H., and Paschalidis, I. (2019). Learning policies for markov decision processes from data. *IEEE Transactions on Automatic Control*, 64, 2298–2309.

Herzallah, R. (2015). Fully probabilistic control for stochastic nonlinear control systems with input dependent noise. *Neural networks*, 63, 199–207. doi:10.1016/j.neunet.2014.12.004.

Kárný, M. (1996). Towards fully probabilistic control design. *Automatica*, 32(12), 1719–1722. doi:10.1016/s0005-1098(96)80009-4.

Kárný, M. and Guy, T.V. (2006). Fully probabilistic control design. *Systems & Control Letters*, 55(4), 259–265. doi:10.1016/j.sysconle.2005.08.001.

Kullback, S. and Leibler, R. (1951). On information and sufficiency. *Annals of Mathematical Statistics*, 22, 79–87.

Krn, M. and Kroupa, T. (2012). Axiomatisation of fully probabilistic design. *Information Sciences*, 186(1), 105 – 113. doi:https://doi.org/10.1016/j.ins.2011.09.018.

Pegueroles, B.G. and Russo, G. (2019). On robust stability of fully probabilistic control with respect to data-driven model uncertainties. In *2019 18th European Control Conference (ECC)*, 2460–2465. doi:10.23919/ECC.2019.8795901.

Peterka, V. (1981). Bayesian approach to system identification. 239–304. doi:10.1016/b978-0-08-025683-2.50013-2.

Quinn, A., Kárnỳ, M., and Guy, T.V. (2016). Fully probabilistic design of hierarchical bayesian models. *Information Sciences*, 369, 532–547. doi:10.1016/j.ins.2016.07.035.

Ramachandran, D. and Amir, E. (2007). Bayesian inverse reinforcement learning. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, 2586–2591. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Ratliff, N.D., Silver, D., and Bagnell, J.A. (2009). Learning to search: Functional gradient techniques for imitation learning. *Autonomous Robots*, 27(1), 25–53.

Wabersich, K.P. and Zeilinger, M.N. (2018). Scalable synthesis of safety certificates from data with application to learning-based control. In *2018 European Control Conference (ECC)*, 1691–1697. doi:10.23919/ECC.2018.8550288.

Xu, T. and Paschalidis, I.C. (2019). Learning models for writing better doctor prescriptions. In *2019 18th European Control Conference (ECC)*, 2454–2459. doi:10.23919/ECC.2019.8796280.

Ziebart, B.D., Maas, A., Bagnell, J.A., and Dey, A.K. (2008). Maximum entropy inverse reinforcement learning. In *Proc. AAAI*, 1433–1438.

[1] See `https://arxiv.org/abs/2001.04428` for a sketch of the proofs.