Data-Efficient Reinforcement Learning for Energy Optimization of Power-Assisted Wheelchairs

Guoxi Feng, Lucian Buşoniu, Thierry-Marie Guerra, Sami Mohammad

***Abstract*—The objective of this paper is to develop a method for assisting users to push Power-Assisted Wheelchairs (PAW) in such a way that the electrical energy consumption over a predefined distance-to-go is optimal, while at the same time bringing users to a desired fatigue level. This assistive task is formulated as an optimal control problem and solved in [17] by the model-free approach Gradient Partially Observable Markov Decision Processes. To increase the data efficiency of the model-free framework, we propose here to use Policy learning by Weighting Exploration with the Returns (PoWER) with 25 controller parameters. Moreover, we provide a new near-optimality analysis of the finite-horizon fuzzy Q-iteration, which derives a model-based baseline solution to verify numerically the near-optimality of the presented model-free approaches. Simulation results show that the PoWER algorithm with the new parameterization converges to a near-optimal solution within 200 trials and possesses the adaptability to cope with changes of the human fatigue dynamics. Finally, 24 experimental trials are carried out on the PAW system, with fatigue feedback provided by the user via a joystick. The performance tends to increase gradually after learning. The results obtained demonstrate the effectiveness and the feasibility of PoWER in our application.**

***Index Terms*—Assistive control, disabled persons, power-assisted wheelchairs, reinforcement learning.**

I. Introduction

Power-assisted wheelchairs (PAWs) are becoming one of the most used tools for disabled persons in ageing societies [1]. One important characteristic of PAWs is that users can perform a tunable and suitable level of physical activities which could not be achieved with traditional manual wheelchairs or fully electric wheelchairs. Moreover, PAWs are driven by a hybrid energy source consisting of human metabolic power and electrical power from a battery. Thanks to this hybrid energy storage structure of PAWs, more degrees of freedom are available to design an optimal energy management strategy.

In this context, the major novelty of this paper is a reinforcement learning control strategy for PAWs that optimizes electrical energy while also taking into account human fatigue. We formulate the assistive task as a constrained optimal control problem: the assistive algorithm is expected to produce a desired fatigue variation of users while using minimal electrical energy for a given driving task. With the initial-to-final fatigue constraint, a (near-) optimal assistance is found so that users contribute efficiently their metabolic energy.

In contrast to hybrid electrical bicycles [2]-[4], little work is done in the PAW literature to address energy optimization with human fatigue considerations. In [5], a regenerative braking control is applied to PAWs for safe downhill driving and electrical energy savings. In [6], the control system is based on a fuzzy algorithm and the fuzzy rules are designed by an expert, aiming to increase the energy efficiency. However, human fatigue has not been taken into account to design PAWs in the literature. The adaptability of optimal solutions with respect to different human fatigue dynamics is not analyzed. An adaptable solution would be vital for PAW designs, since different users may have different fatigue dynamics. Consequently, the existing model-based approaches would not be appropriate for our PAW energy management problem.

The present study relies on Patent WO2015173094 [7] and designs assistive strategies for paraplegic wheelchair users. Specifically, we propose to use model-free reinforcement learning methods to calculate the optimal assistance while respecting a desired fatigue variation over a prescribed driving task. The optimal control method of choice is the direct policy search Policy Gradient (PG) [8]-[9]. Compared with policy iteration [10] and temporal difference learning [11], PG directly provides continuous actions without computing the value function [12], which renders it more practical in robotics [13]. Another crucial advantage of PG is its online model-free nature: it treats the wheelchair dynamics, human fatigue dynamics, and human controller as a "black box", and the algorithm only needs state measurements and rewards (negative costs) in order to learn the solution. This is important in practice, since the true human dynamics will never be available.

Before moving on to practical implementation, the learning methodology must be evaluated in simulation with mathematical models which can roughly represent the human-wheelchair behaviors. We select the human state of fatigue (S_{of}) model from [14], the human controller model [15] and the wheelchair model from [16] and use these models to verify

Manuscript received August 31, 2018; revised December 28, 2018; accepted February 14, 2019. This work was supported by the French Ministry of Research and the grant of the Romanian Ministry of Research and Innovation, CNCS - UEFISCDI, project number PN-III-P1-1.1-TE-2016-0670, within PNCDI III.

Guoxi Feng, Thierry-Marie Guerra are with LAMIH UMR CNRS 8201, Université Polytechnique Hauts-de-France, F-59313 Valenciennes, France (guoxi.feng@uphf.fr, thierry.guerra@uphf.fr).

Lucian Buşoniu is with the Department of Automation, Technical University of Cluj-Napoca, Memorandumului 28, 400114 Cluj-Napoca, Romania (lucian@busoniu.net).

Sami Mohammad is with AutoNomad Mobility, Le Mont Houy, F-59313 Valenciennes 9, France (sami.mohammad@autonomad-mobility.com).

numerically the optimality of the solution found by PG. The baseline solution is given by a finite-horizon extension of the fuzzy Q-iteration in [18]. The two PG methods used in this paper are Gradient of a Partially Observable Markov Decision Processes (GPOMDP) [19]-[20] and Policy learning by Weighting Exploration with the Returns (PoWER) [21].

Compared to the previous work in [17], here we aim to improve considerably the data efficiency of the approach, by employing a different learning algorithm, PoWER, and by simplifying the parametrization of the controller. The idea is to find a near-optimal policy in much fewer trials, so as make the method better in practice. We also derive a new near-optimality analysis of fuzzy Q-iteration, which is not included in [18]. Moreover, the learning method is expected to be adaptable to either different users or changes in the same user. To verify this possibility, a novel investigation is performed in this paper. We modify the human fatigue dynamics to represent three categories of users: physically strong, normal and weak. Simulations are conducted to confirm if the proposed learning method is able to provide a solution that adapts to these cases. We also study the different convergence speeds to the baseline solution when using the parameters learned with the nominal fatigue model versus resetting them to zero defaults.

Our objective with the simulations described above is to evaluate, as a proof of concept, the effectiveness of the learning methodology in the PAW domain. To this end, we select the coarse models [14]-[16]. While these models do generate qualitatively and physically meaningful interconnected human-wheelchair behaviors [17], and thus are useful as an initial validation step, they are not required to be very accurate. Indeed, the main strength of the learning algorithm is that it does not depend on the details of the particular model or notion of fatigue used, instead working for a wide range of unknown dynamics. Having performed these simulations, our next step is to conduct an experiment with the real PAW, where the fatigue model is replaced by a joystick, using which users return a discrete subjective evaluation of their S_{of} to the learning algorithm (too fatigued, OK, and insufficiently fatigued/desiring more exercise). This experiment serves to verify whether the learning methodology works in the real application, which by necessity is quite different from the simulation model.

This paper is organized as follows. In Section II, we present the human-wheelchair model and the problem formulation. Section III gives background on the optimal control methods applied. In Section IV, we apply PoWER and GPOMDP for PAW design and the model-free performance is compared to the model-based baseline. In Section V, we investigate the adaptability of PoWER to changes in the human fatigue dynamics. Section 0 presents the experimental results. Section 0I gives our conclusion and discusses direction for future work.

List of symbols

t, T_e	Continuous time and sampling time
k	Discrete-time sample
F_h	Human applied force
F_m	Maximum available force
U_h, U	Human applied toque and motor torque
d_{ref}, S_{of-ref}	Desired distance-to-go and desired final state of fatigue
x, u	State vector and control input
ξ	Discrete-time state transition function
τ	Trajectories of states and control actions
R	Return
T, r	Terminal reward and stage reward
K	Finite time horizon
$Q^*, \hat{Q}$	Optimal Q-value and approximate Q-value
ε	Error between Q^* and $\hat{Q}$
π^*	Deterministic optimal control policy
$\hat{\pi}$	Model-based approximated optimal policy
$\bar{\pi}$	Model-free deterministic policy
$\tilde{\pi}$	Model-free stochastic policy
ϕ	Triangular membership function
$\bar{x}_i, \bar{u}_j$	Centre of MF ϕ_i and discrete action j
θ	Model-based control parameters
λ	Model-free control parameters

II. Modelling and Problem statement

Next, we introduce a human-wheelchair model which is used to validate in simulation the proposed model-free PG approaches. The proposed human model represents only coarsely human behaviors in practice, since human muscle fatigue would be difficult to precisely model or quantitatively measure [14]. However, the model is sufficiently representative to validate numerically the learning approach.

A. Human Fatigue Model

Owing to the repetitive nature of wheelchair pushing and the absence of a dynamical human fatigue model dedicated to PAWs in the literature, we apply the muscle fatigue model from [14] used for a cycling application. The chosen single-state human fatigue model takes into account the fatigue effect and the recovery effect which usually happen for long-term sports such as wheelchair pushing [25]. Considering these two effects, an intelligent assistance can be devised to save electrical energy. Although significant differences exist between the bicycle problems and PAW problems, this model is still qualitatively meaningful and therefore useful for numerical validation.

The dynamics of the maximum available force F_m provided by human are:

$$\dot{F}_m(t) = -\left(\mathcal{R} + \frac{\mathcal{F}F_h(t)}{M_{vc}}\right)F_m(t) + RM_{vc} \tag{1}$$

where $0 \le F_h(t) \le F_m(t) \le M_{vc}$, the variable t is time, M_{vc} is the Maximum Voluntary Contraction force at rest, and $F_h(t)$ is the actual human applied force. Moreover, $\mathcal{F}$ and $\mathcal{R}$ represent the fatigue coefficient and the recovery coefficients respectively.

When $F_h = F_m$, F_m decreases at its maximum rate. This leads (1) to an equilibrium point where the fatigue rate is identical to the recovery rate, $\dot{F}_m = 0$, and the positive solution is:

$$F_{eq} = \frac{\mathcal{R}M_{vc}}{2\mathcal{F}}\left(-1 + \sqrt{1 + \frac{4\mathcal{F}}{\mathcal{R}}}\right) \tag{2}$$

This positive solution F_{eq} is also the minimum threshold that $F_m(t)$ can achieve. Thus $F_{eq} \le F_m \le M_{vc}$. Using the first-order Euler's method, a discrete-time version of (1) is:

$$F_{m_{k+1}} = \left[1 - T_e\left(\mathcal{R} + \frac{\mathcal{F}F_{h_k}}{M_{vc}}\right)\right]F_{m_k} + T_e\mathcal{R}M_{vc} \tag{3}$$

with the sampling time T_e. Then, the state of fatigue S_{of} in discrete time is defined as:

$$S_{of_k} = \frac{M_{vc} - F_{m_k}}{M_{vc} - F_{eq}} \tag{4}$$

The S_{of} is therefore the normalized value of F_m and is used as an indicator to quantify the human fatigue.

B. Wheelchair Dynamics and Human Controller

The wheelchair is described by the following one-dimensional dynamics:

$$\begin{bmatrix} d_{k+1} \\ v_{k+1} \end{bmatrix} = A \begin{bmatrix} d_k \\ v_k \end{bmatrix} + B\left(U_k + F_{h_k}\zeta\right) \tag{5}$$

where the system matrix $A \in \mathbb{R}^{2\times 2}$ and the input matrix $B \in \mathbb{R}^{2\times 1}$. The control input is the motor torque U and ζ is the wheel radius of hand-rims. The variables d and v are the wheelchair position and velocity, respectively. Note that the human torque satisfies $U_{h_k} = F_{h_k}\zeta$.

We assume that the human force F_h depends on the fatigue state S_{of}, the electrical motor torque U, and the wheelchair velocity v (all perceived by the user):

$$F_{h_k} = y(U_k, S_{of_k}, v_k) \tag{6}$$

Here, we extend the fatigue-motivation model [15] to describe roughly how the fatigue and the assistance affect human motivation. An accurate model of the motivation would require significant further study, but that is outside the scope of this paper, since it would not contribute significantly to our initial objective of validating the learning methodology with a coarse model. Human fatigue decreases the motivation and the perceived help increases it. The normalized help is:

$$H_k = U_k / U_{max} \in [0,1] \tag{7}$$

where U_{max} is the maximum motor torque. The equilibrium point between the perceived fatigue and the perceived help is:

$$f_k = \frac{H_k - S_{of_k}}{H_k + S_{of_k}} \in [-1,1] \tag{8}$$

The motivation $\mathcal{M}$ is:

$$\mathcal{M}_k = \begin{cases} f(1+f_k) & \text{if } f_k < 0 \\ f + (1-f)f_k & \text{if } f_k \geq 0 \end{cases} \tag{9}$$

where $\mathcal{M} \in [0,1]$ and the parameter $f \in [0,1]$. The user motivation in (9) affects proportionally the desired wheelchair velocity V_r of the user, so that a higher motivation leads to a higher desired velocity, i.e. $V_r = V_{max}\mathcal{M}$ (where V_{max} is the maximum velocity of the wheelchair). Finally, the human force is modeled as a proportional velocity-tracking controller:

$$F_{h_k} = K_p(V_{max}\mathcal{M}_k - v_k) \tag{10}$$

Moreover, the human force should be saturated by F_m, and only positive human force is taken into account:

$$F_{h_k} = \text{sat}(0, F_{m_k}, F_{h_k}) \tag{11}$$

C. Optimal Control Problem Statement

For simplicity, we consider the electric energy consumption to be a quadratic function of U via the finite horizon criterion:

$$\frac{1}{2}\sum_{k=0}^{K-1} U_k^2 \tag{12}$$

Over a predefined time horizon, the optimal solution minimizing (12) without considering any constraint corresponds to a manual propulsion strategy in which all the kinetic energy comes from the human. To avoid this trivial solution, we impose the following fatigue constraint. Knowing the initial S_{of_0}, the final S_{of_K} should reach a desired level S_{of-ref}:

$$S_{of_K} = S_{of-ref} \tag{13}$$

while minimizing (12) over the considered driving profile. The wheelchair should also travel a required distance. Knowing the initial d_0, we impose the following distance constraint:

$$d_K = d_{ref} \tag{14}$$

including the terminal distance d_K and the desired terminal d_{ref}. Rather than solving explicitly a constrained problem, we represent the constraints (13)-(14) with a terminal reward, leading to the following optimal control problem:

$$\max_{U_m} R = -[w_1\ w_2]\begin{bmatrix} \left(d_K - d_{ref}\right)^2 \\ \left(S_{of_K} - S_{of-ref}\right)^2 \end{bmatrix} - \frac{1}{2}\sum_{k=0}^{K-1} U_k^2 \tag{15}$$

with w_1, w_2 the reward weights and K the finite time horizon. Note that in classical control theory the return R in (15) is often replaced by a positive cost function and must be minimized. Here, we use Artificial Intelligence techniques, so we follow the maximization convention in this field.

III. Optimal control algorithm

This section first introduces the finite-horizon version of Fuzzy Q-iteration [18] and a corresponding explicit bound on its suboptimality. This algorithm provides the baseline to which we will compare the existing model-free algorithms, GPOMDP [9] and PoWER [21]. To fully understand the model-free design, these algorithms are presented in Section B.

The system considered is described in general by the deterministic state transition function:

$$x_{k+1} = \xi(x_k, u_k) \tag{16}$$

where x and u are state vector and control input respectively. The general return R to optimize over a finite-horizon is:

$$R(\tau) = \gamma^K T(x_K) + \sum_{k=0}^{K-1} \gamma^k r(x_k, u_k) \tag{17}$$

where $\tau = (x_0, u_0, x_1, u_1, \dots x_{K-1}, u_{K-1}, x_K)$ is a trajectory of the system, $T(x_K)$ is the terminal reward, and $r(x_k, u_k)$ is the stage reward. A discount factor $\gamma \in (0, 1]$ may be used; in the finite-horizon case, γ is often taken equal to 1. The optimization problem (15) is a specific case of the general form (17). The following algorithms are presented for the general case defined in (16)-(17).

A. Finite-Horizon Fuzzy Q-Iteration

Fuzzy Q-iteration [18] is originally given in the infinite-horizon case, and the horizon-K solution can be obtained simply by iterating the algorithm K times. However, the entire time-varying solution must be maintained, and special care must be taken to properly handle the terminal reward. So for clarity we restate the entire algorithm, adapting it to the finite-horizon case.

The idea is to approximate the optimal time-varying solution, which can be expressed using Q-functions of the state in the state-space X and action in the action space U. These Q-functions are generated backwards in time:

$$Q^*_{K-1}(x_{K-1}, u_{K-1}) = r(x_{K-1}, u_{K-1}) + \gamma T\left(\xi(x_{K-1}, u_{K-1})\right)$$

$$Q_k^*(x_k, u_k) = r(x_k, u_k) + \gamma \max_{u_{k+1}} Q_{k+1}^*(\xi(x_k, u_k), u_{k+1}), \quad \text{for } k = K-2, \dots, 0 \text{ and } \forall x \in X, \forall u \in U \tag{18}$$

The advantage of using Q-functions is that the optimal control can then be computed relatively easily, using the following time-varying state-feedback:

$$\pi^*(x_k, k) = \arg\max_{u_k} Q_k^*(x_k, u_k) \tag{19}$$

Since the system is nonlinear and the states and actions are continuous, in general it is impossible to compute the exact solution above. We will therefore represent Q^* with an approximator that relies on an interpolation over the state space, and on a discretization of the action space. First, to handle the action, the approximate Q-value of the pair (x, u) is replaced by that of the pair (x, u_d), where u_d has the closest Euclidean distance to u in a discrete subset of actions $U_d = \{\bar{u}_j | \bar{u}_j \in U, j = 1, \dots, N_u\}$. To handle the state, a grid of discrete values $X_d = \{\bar{x}_i | \bar{x}_i \in X, i = 1, \dots, N_x\}$ in the state space is chosen for the centers of triangular membership functions $\phi(x) = [\phi_1(x), \dots, \phi_{N_x}(x)]$ [18]. A parameter vector $\theta \subset \mathbb{R}^{N_x \times N_u \times K}$ is defined, and the approximate Q-function is linearly interpolated by overlapping the membership functions ϕ on the grid of the centers X_d as follows:

$$\hat{Q}_k(x, u) = \sum_{i=1}^{N_x} \phi_i(x) \theta_{i,j,k} \tag{20}$$

with $j \epsilon \arg\min_{j'} \left\| u - \bar{u}_{j'} \right\|^2$. Thus, each individual parameter corresponds to a combination between a point i on the state interpolation grids, a discrete action j, and a time stage k. The approximated optimal solution $\hat{\pi}$ can be obtained as follows:

$$\hat{\pi}(x, k) = \bar{u}_j \text{ with } j = \arg\max_{j'} \textstyle\sum_{i=1}^{N_x} \phi_i(x) \theta_{i,j',k}. \tag{21}$$

Algorithm 1. Finite-horizon fuzzy Q-iteration

```
1 for $i = 1, \dots, N_x, j = 1, \dots, N_u$ do
2   $\theta_{i,j,K-1} = r(\bar{x}_i, \bar{u}_j) + \gamma T\left(\xi(\bar{x}_i, \bar{u}_j)\right)$
3 end for
4 for $k = K-2, \dots, 0$ do
5     for $i = 1, \dots, N_x, j = 1, \dots, N_u$ do
6   $\theta_{i,j,k} = r(\bar{x}_i, \bar{u}_j) + \gamma \max_{j'} \sum_{i'=1}^{N_x} \phi_{i'}(\xi(\bar{x}_{i'}, \bar{u}_{j'}))\theta_{i,j,k+1}$
7     end for
8   $\hat{\pi}(x, k) = \bar{u}_j,\ j = \arg\max_{j'} \sum_{i=1}^{N_x} \phi_i(x) \theta_{i,j',k} \quad \forall x, k$
9 end for
```

Algorithm 1 gives the complete version of Fuzzy Q-iteration. To understand it, note that the main update in line 6 is equivalent to the following approximate variant of the iterative update in (18):

$$\hat{Q}_k(\bar{x}_i, \bar{u}_j) = r(\bar{x}_i, \bar{u}_j) + \gamma \max_{\bar{u}_{j,k+1}} \hat{Q}_{k+1}(\xi(\bar{x}_i, \bar{u}_j), \bar{u}_{j,k+1})$$

This is because, firstly, due to the properties of triangular basis functions the parameter $\theta_{i,j,k}$ is equal to the approximate Q-value $\hat{Q}_k(\bar{x}_i, \bar{u}_j)$. Secondly, the maximization over the discretized actions is done by enumeration over j; and thirdly, the summation is just the approximate Q-value at the next step, via (20). Line 2 simply sets the parameters at step K-1 via the initialization in (18).

For clarity, the algorithm shows in line 8 how the near-optimal control is computed via maximization over the discrete actions. In practice, this maximization is done on-demand, only for the states encountered while controlling the system, so an explicit function $\hat{\pi}$ of the continuous state does not have to be stored. Instead, only the parameters are stored.

In contrast to the algorithm itself, the infinite-horizon analysis does not easily extend to the finite-horizon case, e.g. we need to account for the possibility that $\gamma = 1$. Thus, the upcoming analysis is a novel contribution of the present paper. Due to space limitations, we give it here without proofs, which can be found in the supplementary material at: http://busoniu.net/files/papers/tie_suppl.pdf.

The error ε_k between $\hat{Q}_k$ and Q_k^* for time k is defined as:

$$\varepsilon_k = \left\| \hat{Q}_k(x, u) - Q_k^*(x, u) \right\| \tag{22}$$

The state resolution step δ_x is defined as the largest distance between any two neighboring triangular MF cores, i.e.

$$\delta_x = \max_{i \in \{1, \dots N_x\}} \min_{i' \in \{1, \dots N_x\}, i' \neq i} \|\bar{x}_i - \bar{x}_{i'}\|_2 \tag{23}$$

The action resolution step δ_u is defined similarly for the discrete actions. Moreover, for every x, only $2^{N_{state}}$ (where N_{state} is the number of states) triangular membership functions are activated. Let the infinite norm $\|\theta_k\|_\infty = \max_{i \in \{1, \dots N_x\}, j \in \{1, \dots N_u\}} |\theta_{i,j,k}|$ denotes the largest parameter magnitude at sample k. Note that triangular membership functions are Lipschitz-continuous, so there exists a Lipschitz constant $L_\phi > 0$ such that $\|\phi_i(x) - \phi_i(x')\|_2 \le L_{\phi_i}(\|x - x'\|_2)$ $\forall x, x' \in X, \forall i$. Moreover, we say that a function of the state and action, such as the deterministic state transition function ξ, is Lipschitz continuous with constant $L_\xi > 0$ if $\|\xi(x, u) - \xi(x', u')\|_2 \le L_\xi(\|x - x'\|_2 + \|u - u'\|_2)$ $\forall x, x' \in X, u, u' \in U$.

Assumption 1: We assume that the reward function r, the terminal function T, and the deterministic state transition function ξ are Lipschitz-continuous with the Lipschitz constants L_r, L_T, and L_ξ respectively.

We present an explicit bound on the near-optimality of the Q-function as a function of the grid resolutions. This bound has the nice feature that it converges to zero when the grid becomes infinitely dense, which is a consistency property of the algorithm.

Proposition 1: Under Assumption 1, there exists an error bound $\bar{\varepsilon}_k$ so that $\hat{Q}$, i.e. the approximate Q-function obtained by (22) satisfies $\varepsilon_k \le \bar{\varepsilon}_k$ and $\lim_{\delta_x, \delta_u \to 0} \bar{\varepsilon}_k = 0$ for $k = K-1, \dots, 0$. Depending on the discount factor γ and the Lipschitz constant L_ξ, the bound is given as follows:

1) When $\gamma L_\xi < 1$, $\bar{\varepsilon}_k = (K-k)L_r(\delta_x + \delta_u) + \sum_{z=1}^{K-k}\left(L_T(\gamma L_\xi)^z + L_r \frac{(\gamma L_\xi)^z - \gamma L_\xi}{\gamma L_\xi - 1}\right)(\delta_x + \delta_u)$.
2) When $\gamma L_\xi = 1$, $\bar{\varepsilon}_k = (K-k)(L_r + L_T)(\delta_x + \delta_u) + \frac{(K-k)(K-k-1)}{2} L_r(\delta_x + \delta_u)$.
3) When $\gamma L_\xi > 1$, $\bar{\varepsilon}_k = (K-k)L_r(\delta_x + \delta_u) + 2^{N_{state}} \gamma L_\xi L_\phi(\delta_x + \delta_u) \sum_{z=1}^{K-k} \|\theta_{K-z+1}\|_\infty$.

B. Policy Gradient Algorithms: GPOMDP and PoWER

In model-free policy search, exploration is indispensable to learn the unknown dynamics. Stochastic policies are needed to explore. To this end, we use a parameterized policy with the parameters λ. Then, the stochastic policy distribution is $\tilde{\pi}_\lambda(u_k|x_k,k)$. Under this stochastic policy, the probability distribution $p_\lambda(\tau)$ over trajectories τ can be expressed in the following way:

$$p_\lambda(\tau) = p(x_0)\prod_{k=0}^{K-1}\tilde{\pi}_\lambda(u_k|x_k,k) \tag{24}$$

where $p(x_0)$ is the initial state distribution. Under trajectories τ generated by $\tilde{\pi}_\lambda$, the expected return is:

$$\bar{R}_\lambda = \int p_\lambda(\tau)R(\tau)d\tau \tag{25}$$

The GPOMDP (Gradient of a Partially Observable Markov Decision Processes) algorithm [20] updates the control parameters λ in the steepest ascent direction so that the expected return (25) is maximized. We apply this algorithm to estimate the gradient $\nabla_\lambda\bar{R}_\lambda$, which can be obtained from the stage rewards r_j and the distribution $\tilde{\pi}_\lambda$. The entire procedure is given in Algorithm 2, where Γ is the total trials. In line 3 of Algorithm 2, for each iteration l we generate N_τ trajectories using the stochastic policy with λ_l. Applying the Likelihood Ratio Estimator, calculating the gradient $\nabla_\lambda\bar{R}_\lambda$is transformed to calculating $\nabla_\lambda \log\tilde{\pi}_\lambda(u_k|x_k,k)$ (more details can be found in [22], [20]). The stochastic policy distribution $\tilde{\pi}_\lambda$ is available, so that the gradient $\nabla_\lambda\bar{R}_\lambda$ can be computed. The expected value is approximated by Monte Carlo techniques using the N_τ trajectories. The learning rate α has to be tuned manually in order for the control parameters λ to converge efficiently.

Algorithm 2. GPOMDP

```
1 Initialize $\lambda_0$
2 for $l = 0,1,2,\dots\,\Gamma$
3    Generate $N_\tau$ trajectories $\tau$ of length $K$ using $\lambda_l$
4    $\nabla_\lambda\bar{R}_\lambda = \frac{1}{N_\tau}\sum_{\varsigma=1}^{N_\tau}\left[\sum_{k=0}^{K-1}\sum_{h=0}^{k}\left[\nabla_\lambda \log\tilde{\pi}_\lambda(u_k^\varsigma|x_k^\varsigma,k)\right]r_h^\varsigma\right]$
5    $\lambda_{l+1} = \lambda_l + \alpha\cdot\nabla_\lambda\bar{R}_\lambda$with the learning rate $\alpha > 0$
6 end for
```

To obtain a higher expected return, we may consider a new distribution $p_{\lambda'}(\tau)$ over trajectories that might provide a better expected return than the previous one i.e. $\int p_{\lambda'}(\tau)R(\tau)d\tau \geq \int p_\lambda(\tau)R(\tau)d\tau$. The new expected return $\int p_{\lambda'}(\tau)R(\tau)d\tau$ with parameters λ′ is lower-bounded by a quantity $L_\lambda(\lambda')$ that depends on λ. The analytical expression of $L_\lambda(\lambda')$ can be found in [21]. The selection of λ′ can be done by maximizing the lower bound $L_\lambda(\lambda')$ to implicitly maximize (25). In [23], the authors show that maximizing $L_\lambda(\lambda')$ guarantees the improvement of the expected return. The intuition is that if $R(\tau_1) > R(\tau_2)$, the new λ′ will put more probability mass on τ_1 than λ does.

PoWER (Policy learning by Weighting Exploration with the Returns) works by maximizing the lower bound $L_\lambda(\lambda')$. Moreover, a deterministic policy is approximated by general basis functions ψ i.e. $\bar{\pi}_\lambda(x_k) = \lambda^T\psi(x_k,k)$. To explore, Gaussian noise is added directly to the parameter vector λ. Using importance sampling, the parameters λ are updated with the N_s trials which have the highest return among the performed trials. The formula to update the parameters is [21]:

$$\lambda_{l+1} = \lambda_l + \frac{\sum_{s=1}^{N_s}(\lambda_s - \lambda_l)R(\tau_s)}{\sum_{s=1}^{N_s}R(\tau_s)} \tag{26}$$

The whole method is given in Algorithm 3.

Algorithm 3. PoWER

```
1 Initialize $\lambda_0$
2 for $l = 0,1,2,\dots\,\Gamma$
3    Generate a new trajectory $\tau$ of length $K$ using $\lambda_l$
4    Sort the performed trials decreasingly by return
5    Select the $N_s$ trials with the highest return
6    Update parameters $\lambda_{l+1} = \lambda_l + \frac{\sum_{s=1}^{N_s}(\lambda_s - \lambda_l)R(\tau_s)}{\sum_{s=1}^{N_s}R(\tau_s)}$
7 end for
```

IV. Data-efficient policy gradient learning for the assistance problem

The energy optimization problem has been solved in [17] by using a considerable amount of data, which is expensive to obtain in practice. In this section, the main purpose is to increase the data efficiency. To achieve this goal, we propose two ideas. The first one is to use a more efficient PG algorithm, namely PoWER. Secondly, as observed in [17], the operating region in the state space is concentrated on a few radial basis functions (RBFs); therefore, for the remaining RBFs the parameters remain constant or have a very small gradient. Reducing the parameters to the significant ones will accelerate the learning speed. Using Fuzzy Q-iteration as the baseline solution, we compare the performance of the two PG algorithms (PoWER and GPOMDP) with the controller parameterizations (29) and the one in [17]. In addition, we give an analysis of the policies obtained to explain how the assistive torques maximize the return in (15).

To represent the problem (15), the terminal reward and the stage reward of (17) are defined as follows:

$$T(x_N) = -[w_1\, w_2]\begin{bmatrix}(d_K - d_{ref})^2\\ \left(S_{of_K} - S_{of-ref}\right)^2\end{bmatrix} \tag{27}$$

$$r(x_k,u_k) = -\frac{1}{2}U_k^2 \tag{28}$$

where the state vector is $x_k = \left[d_k, v_k, S_{of_k}\right]^T$ and the control input is the motor torque $u_k = U_k$.

Since the driving task is to travel a predefined distance, negative human torque and negative motor torque are inefficient in terms of metabolic-electrical energy consumption over the driving task. Moreover, due to the actuator limitations, the maximum torque that the motor can provide is U_{max}. Therefore, the control is bounded: $0 \leq U \leq U_{max}$. Since the distance is monotonic, it acts as a proxy for time, which can be implicitly used by the algorithm instead of an explicit time variable. Therefore, we can use a time-invariant solution $\bar{\pi}_\lambda(x_k)$ to approximate the optimal time-varying solution in (19). We approximate the deterministic part $\bar{\pi}$ of the motor torque by the following RBF expansion:

$$\bar{\pi}_\lambda(x_k) = \lambda_l^T\varphi(x_k) \tag{29}$$

where the RBF $\varphi_i = \exp(-\beta\|x_k - c_i\|^2)$, $c_{i=1,\dots,M}$ is the center vector of the RBFs, M is the total number of RBFs and β is the radial parameter. Since the radial parameter β is the same for each RBF, all the RBFs have the same shape. PoWER and GPOMDP for the Assistance Problem.

In the rest of the paper, for each variable, a subscript or index P (resp. G) stands for PoWER (resp. GPOMDP).

For PoWER, the general basis functions ψ in Algorithm 3 are replaced by the RBFs φ in (29). The exploration is carried out in the parameter space as previously explained. The zero mean Gaussian noise vector z_P with the standard deviation σ_P is added to the parameters and renders the action stochastic as follows:

$$\operatorname{sat}\left(0, U_{max}, (\lambda_l^P + z_P)^T\varphi(x_k)\right) \tag{30}$$

where the stochastic motor torque is saturated between 0 and U_{max} and the parameter vector λ_l^P is updated by (26).

Regarding GPOMDP, zero mean Gaussian noise z_G is added to the executed action and renders the policy (29) stochastic. In order to prevent the executed action from violating the action saturation limits, the stochastic motor torque is selected with:

$$q_{\text{sat}}\left[\lambda_l^{G^T}\varphi(x_k) + z_G\right] \tag{31}$$

where q_{sat} is a smooth saturation (the Gaussian error function [24] shown at the top of Fig. 1) between $[0, U_{max}]$ such that the stochastic action is differentiable with respect to λ_l^G. When the optimal action is close to the borders of the interval $[0, U_{max}]$, using the original return (28) without input saturation can lead to the divergence of the parameters. To address this problem, a penalty function P is added to the stage reward (28) as follows:

$$r(x_k, u_k) = -\left[\frac{1}{2}U_k^2 + w_3 P(U_k)\right] \tag{32}$$

where w_3 is the constraint penalty weight. The function P, shown in Fig. 1 bottom, is defined as follows:

$$P = \begin{cases} \sin\left(\dfrac{\pi\cdot(U - U_{max})}{0.04U_{max}}\right) + 1 & 0.98U_{max} \le U \le U_{max} \\ 0 & 0.02U_{max} \le U \le 0.98U_{max} \\ \sin\left(\dfrac{\pi\cdot(-U)}{0.04U_{max}}\right) + 1 & 0 \le U \le 0.02U_{max} \end{cases} \tag{33}$$

which penalizes the (stochastic) action when it is close to the saturation value. The objective of P is to keep the mean value of the stochastic actions inside the interval $[0, U_{max}]$.

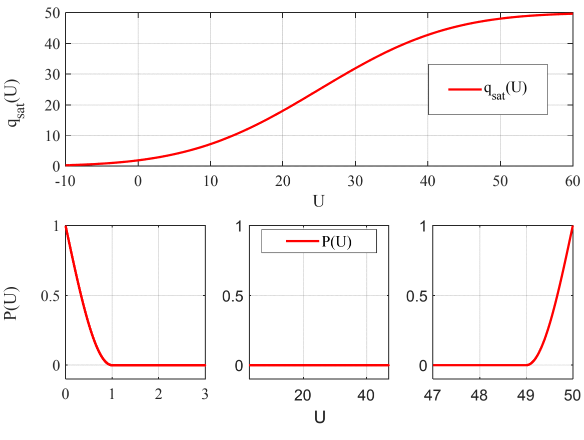


Figure 2: Smooth saturation function q_{sat} (above) and penalty function P_s for $U_{max} = 50N$ (below)

Recall that we use a time-invariant policy. Consequently, the stochastic action distribution does not depend on the time stage k, but on the state x_k. According to (31), the distribution $\tilde{\pi}_\lambda^G(U_k|x_k)$ of the stochastic motor torque U is:

$$\tilde{\pi}_\lambda^G(U_k|x_k) = \frac{1}{\sqrt{2\pi\sigma_G^2}}\exp\left(-\frac{\left[q_{\text{sat}}{}^{-1}(U_k) - \lambda_l^{G^T}\varphi(x_k)\right]^2}{2\sigma_G^2}\right) \tag{34}$$

The derivative of (34) with respect to λ_l^G is used to estimate the gradient $\nabla_\lambda \bar{R}_\lambda$ in Algorithm 2 and to update the parameter vector λ_l^G.

By tuning the parameters (β, c, M) of the basis functions (29), the standard deviation σ_P and σ_G, the reward weights (w_1, w_2), the learning rate α and the penalty weight w_3, we have all the conditions to update the parameters λ^P or λ^G.

A. Results with PoWER and GPOMDP

In this section, simulations are carried out to compare the proposed methods. The whole set of parameters is shown in Table I. The human model parameters are adapted from [14] to have a reasonable fatigue and recovery rate, which render the optimal more challenging and avoid a trivial optimal solution. The control strategy is approximated over the state-space and action-space region given in Table I. The configurations and learning parameters of the return function, penalty function, model-based policy, and model-free policies are shown in Table II later.

For the simulation in Fig. 2, the exploration noise (and the learning rate for GPOMDP) are tuned to deliver a good performance for each configuration and each algorithm. The number in the legend gives the total parameters of the controller approximation (29) for each simulation. A mean value along with a 95% confidence interval calculated for 10 independent simulations is given (each simulation with 400 trials). Fig. 2 shows that with the same policy parametrization, PoWER has a considerably higher data efficiency than GPOMDP. GPOMDP-25 and GPOMDP-200 give a similar final performance. Considering the mean, 90% of the baseline return is provided in around 100 trials by PoWER-25. The same performance is

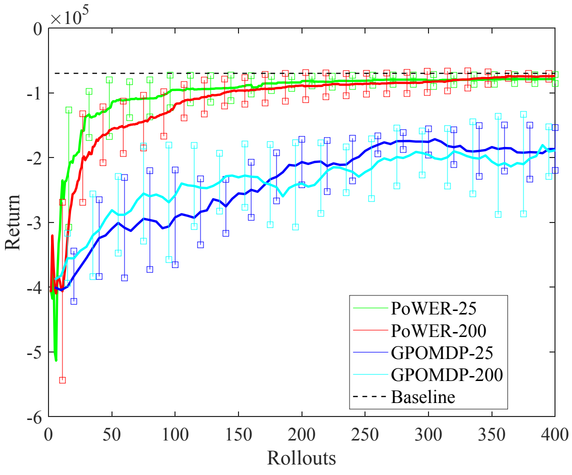


Figure 1: The mean performance and 95% confidence interval on the mean value of PoWER with 25 control parameters (PoWER-25), PoWER with 200 control parameters (PoWER-200), GPOMDP with 25 control parameters (GPOMDP-25) and GPOMDP with 200 control parameters (GPOMDP-200)

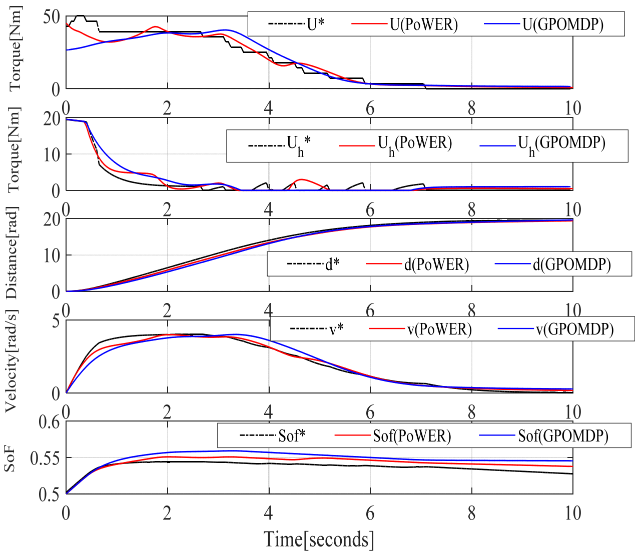


Figure 3: Controlled trajectories provided by Policy Gradient algorithms and fuzzy Q-iteration algorithm

given in around 200 trials by PoWER-200. Overall, PoWER-25 is the best choice among the considered configurations.

For the next simulations, we focus on the final near-optimal behaviours provided by PoWER-25 and GPOMDP-25. To this end, 400 trials and 8000 trials are performed to learn the parameter vectors λ^P and λ^G, respectively. The slow learning speed of GPOMDP is mainly due to the exploration noise added directly to actions at every step. This type of exploration strategy can cause a high variance for learning algorithm [21] and leads to a poor performance in terms of data-efficiency. As shown in Fig. 3, the model-free methods PoWER (red solid line) and GPOMDP (blue solid line) are comparable to the model-based fuzzy Q-iteration (black dotted line). The final return is -82017, -70242, and -69837 for GPOMDP, PoWER and fuzzy Q-iteration respectively. Here again, PoWER delivers a better solution than GPOMDP in terms of final return.

The simulation was done on an Intel Core i7-6500 CPU @ 2.50GHz. The average elapsed CPU time to compute a control action is $1.0007 * 10^{-4}$s, $7.6464 * 10^{-4}$s, $1.4562 * 10^{-4}$s, and $7.5934 * 10^{-4}$s respectively for PoWER-25, PoWER-200 GPOMDP-25 and GPOMDP-200. As their elapsed CPU time is significantly less than the sampling time of 0.05s, it is possible to embed them into a real PAW.

Remark 2: The RBF grid used for policy approximation has only one center on the S_{of} state for the policy approximation. It may appear surprising that the controller works despite this limitation. Indeed, we have tested it and it works well for initial S_{of} in the interval [0.4, 0.6]. This is because the solutions are similar for this range of S_{of}, as confirmed by the baseline fuzzy Q-iteration. In additional PG experiments with three centers on the S_{of} state and reduced radius in this dimension, performance increased marginally but at the cost of less reliable convergence. Another possible reason is that much of the optimal control input may be open-loop, and again since the distance is monotonic, it acts as a proxy for time. Even if this were so, state feedback is nevertheless crucial in practice to defend against disturbances, so we choose to apply closed-loop, state-feedback policies.

In the beginning, the motor provides a large assistance and the user pushes "hard" to start the wheelchair. After reaching a suitable velocity, the user reduces his applied force to recover. In the remainder of the task, the motor assistance is reduced gradually to minimize the energy consumption while the user continues recovering. The assistance provided tries to use as little electrical energy as possible and enables the user to reach the desired final fatigue state.

PoWER and GPOMDP have a similar terminal error of nearly 0.05 between the final S_{of_K} and the desired final value

TABLE I
PARAMETERS OF THE CONSIDERED HUMAN-WHEELCHAIR DYNAMICS

Meaning	Notation [units]	Value or domain
Sampling time	T_e [s]	0.05
	Human parameters	
Recovery coefficient	$\mathcal{R}[s^{-1}]$	0.0063
Fatigue coefficient	$\mathcal{F}$ $[s^{-1}]$	0.153
MVC	M_{vc} [N]	100
Fraction of V_{max}	f	0.5
Human control gain	K_p	30
	Wheelchair parameters	
Wheel radius	ζ [m]	0.33
Maximum velocity	V_{max} [rad/s]	7
System matrix	A	$\begin{bmatrix}1 & 0.05\\0 & 0.9406\end{bmatrix}$
Input matrix	B	$\begin{bmatrix}0\\0.0059\end{bmatrix}$
	Driving schedule configuration	
Finite horizon	K	200
Initial state of fatigue	S_{of_0}	0.5
Desired final human fatigue	S_{of-ref}	0.5
Distance-to-go	d_{ref} [rad]	20
	State-space and action-space region	
Distance	d [rad]	[0,20]
Velocity	v [rad/s]	[0,7]
State of fatigue	S_{of}	[0.35,0.7]
Motor torque	U [Nm]	[0,50] ($U_{max} = 50$)

TABLE II
RETURN FUNCTION, PENALTY FUNCTION, MODEL-BASED POLICY, MODEL-FREE POLICIES CONFIGURATIONS, AND LEARNING PARAMETERS

Return function and penalty function configuration	
Reward weight matrix $[w_1 \quad w_2]$	$[4000 \quad 10^7]$
Penalty weight w_3	800
Q-function approximation	
Centers of triangular functions ϕ distributed on an equidistant grid	$10 \times 10 \times 41$ over the state-space $\left(x = [d, v, S_{of}]^T\right)$
Number of equidistant discrete actions	15
Radial basis functions (29) configuration 1	
Radial parameter β	0.5
Centers of RBFs distributed on an equidistant grid	$5 \times 5 \times 8$
Total number of RBFs M	200
Radial basis functions (29) configuration 2	
Radial parameter β	0.5
Centers of RBFs distributed on an equidistant grid	$5 \times 5 \times 1$
Total number of RBFs M	25
GPOMDP parameters	
Learning rate α	10^{-5}
Standard deviation σ_G	5
PoWER parameters	
Importance sampling N_s	10
Standard deviation σ_P	1

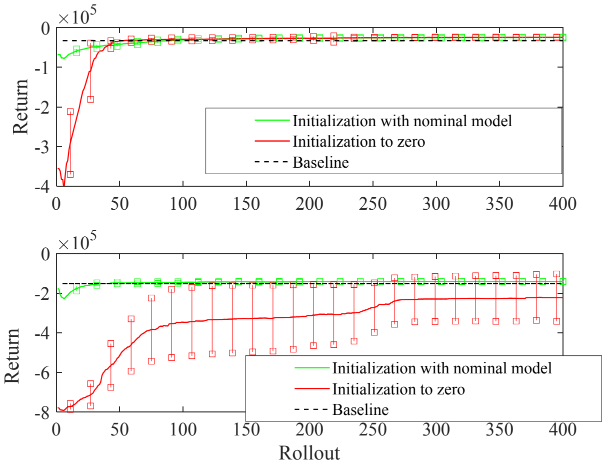


Figure 4: The mean performance of PoWER for both initialization (Top: *η*=2 and bottom *η*=1/2)

S_{of-ref}(compared to 0.02 for fuzzy Q-iteration). This error may be reduced by further tuning the parameters.

V. Adaptability to Changes in the Human Fatigue Dynamics

In this section, we turn our focus towards adaptation to human fatigue variability, which is crucial for a personalized PAW. In what follows, we investigate only the adaptability of PoWER-25 to these changes, since it provided the best results in the previous section. The objective of this investigation is to confirm the possibility of having a generic solution for different human fatigue dynamics. To represent various human fatigue dynamics, we change the parameters of (1) as follows:

$$\mathcal{F}' = \frac{1}{\eta}\mathcal{F}; \quad \mathcal{R}' = \eta\mathcal{R}; \quad M'_{vc} = \eta M_{vc}$$

where $\mathcal{F}$, $\mathcal{R}$, M_{vc} are the nominal parameters used in Section IV. A value $\eta > 1$ corresponds to a user physically stronger than the nominal one, because they get exhausted slower, recover faster and have more Maximum Voluntary Contraction force. On the contrary, $\eta < 1$ corresponds to a physically weaker user. Adaptation starts from the parameters found using the nominal model. As a baseline, we compare this adaptation procedure with simply resetting the parameters to zero values when the model changes. The same variance σ_P of Section IV is applied for exploration.

Both stronger $(\eta = 2)$ and weaker $(\eta = 1/2)$ users are studied. Fig. 4 shows that PoWER is clearly much more efficient, when initialized with the nominal model, being able to provide a good return directly and to find a new near-optimal solution for the new fatigue dynamics in less than 50 trials.

In order to verify whether the assistive control can adapt to a bigger range of parameter changes, we carry out the same comparison for $\eta = 8, 4, 3,\ 1/3,\ 1/4,\ 1/8$. Table III gives the baseline return for each η, the minimal return for each case and the number of trials to converge to 90% of the corresponding baseline return for both initializations. The asterisk * represents situations where the learning algorithm fails to converge to 90% of the baseline return within 400 trials.

TABLE III
PoWER with varying fatigue model (Zero: Initialization to zero, Nominal: Initialization with the nominal model. The minimal return is normalized by the corresponding baseline return)

η	Baseline return (fuzzy Q-iteration)	PoWER			
		Minimal return		Number of trials	
		Nominal	Zero	Nominal	Zero
8	-361950	1.25	2.30	39	37
4	-96723	1.76	4.96	33	56
3	-54018	2.14	7.58	47	34
2	-32744	2.38	12.43	48	65
1/2	-150920	1.51	5.25	10	*
1/3	-207400	1.86	5.52	198	*
1/4	-299540	1.76	4.50	37	*
1/8	-657620	1.56	2.73	30	*

Table III shows that both initializations have similar convergence for $\eta = 8$. For $\eta = 3$, the initialization to zero has a faster convergence. This result may be because that the initialization to zero is closer to the optimal solution. Nevertheless, for all the other η, the initialization with the nominal model converges faster. Overall, starting learning with the nominal solution can guarantee a higher minimum return. Moreover, PoWER with prior knowledge adapts reasonably well to human fatigue dynamics changes without tuning again the learning parameter σ. This study therefore confirms the possibility of providing an adaptive solution for different human fatigue dynamics.

VI. Real-time Experiment

To demonstrate the effectiveness of the proposed learning algorithm, proof-of-concept experiments have been conducted on our PAW prototype, which is equipped with two torque sensors, two position encoders and a joystick. Via the joystick, the user can return a subjective evaluation of their S_{of} to the control algorithm. When the user pushes the joystick to the negative or positive Y-direction, the joystick returns to the algorithm a discrete value -1 or 1, respectively. The neutral position of the joystick returns a discrete value 0. These three discrete values -1, 0 and 1 mean respectively that the user feels too tired, is comfortable, and feels insufficiently tired (is willing to exercise more). The discrete signal is filtered so that when it changes between two levels (among -1, 0 and 1), its filtered version *I* provides a gradual transition between these levels. Furthermore, to avoid the need for too many pushes of the joystick, after such a transition the filtered signal is kept nearly constant for a certain duration.

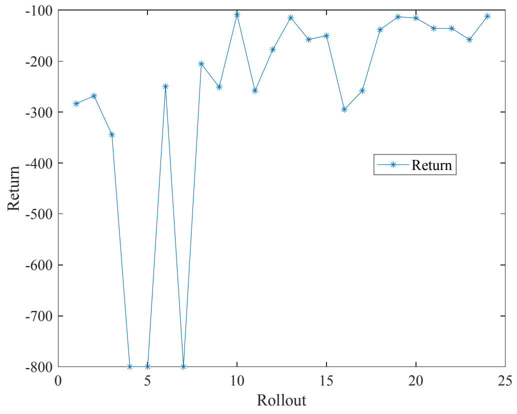


Figure 5: The total return of each trial

The driving scenario consists in riding on a straight flat road with a given reference velocity v_{ref} set by the user. The velocity estimated from the position encoders is available via the computer connected to the data acquisition system. The control objective is to minimize both the electrical energy and the use of the joystick, while tracking the reference velocity. Therefore, the stage reward function is:

$$r_k = -w_1\left(v_k - v_{ref_k}\right)^2 - w_2 I_k^2 - w_3 U_k^2 \tag{35}$$

where v_{ref_k} is the given reference velocity at discrete sample k. The reward weights are $w_1 = 10, w_2 = 0.25$ and $w_3 = 0.05$. Note that any joystick signal $I \neq 0$ is penalized. The controller is configured as a PI-type law:

$$U_k = \lambda_1\left(v_k - v_{ref_k}\right) + \lambda_2 \sum_{\mathrm{i}=0}^{k}\left(v_{\mathrm{i}} - v_{ref_{\mathrm{i}}}\right) + \lambda_3 I_k + \lambda_4 \sum_{\mathrm{i}=0}^{k} I_{\mathrm{i}} - \lambda_5 F_{h_k}\zeta \tag{36}$$

The first four terms of the controller (36) are used to track the reference values v_{ref} whilst keeping the filtered joystick signal I to 0. The term $\lambda_5 F_{h_k}\zeta$ is for compensating the human input.

One healthy male volunteer (29-year-old) performed the proof-of-concept experiments. There are 5-minute rest periods between consecutive trials. In total, 24 trials with the same driving condition have been carried out on the same day to learn the parameter vector λ in (35). Fig. 5 shows the total return of each trial. Among the 24 trials, 3 trials went unstable at the beginning of learning. For these trials, the user stopped immediately the wheelchair and a very low return was given to the learning algorithm to avoid such situations in the future. The return tends to increase gradually after performing these trials. We notice that the obtained curve of return is noisy. Due to the time-consuming nature of the experiment, it is not feasible to perform many trials to obtain a smooth mean return. Therefore, we analyze qualitatively the obtained trajectories.

Fig. 6 shows the trajectories of the first four stable trials and the last four trials. We remark that the user does not push the joystick anymore in the last four trials. The joystick signal I sums up the influence of main physiological and psychological factors to tell the learning algorithm what assistive torque is suitable to users. The fact that the user does not use anymore the joystick means that after training, the provided assistive torques are acceptable in terms of the sensation of fatigue. Another consequence of training is that the user and the controller track together the given velocity more smoothly.

Through these proof-of-concept experiments, we conclude that the proposed learning algorithm PoWER is able to improve the performance of the controller (36). For a final commercial product, there will be a certain accommodation time to obtain a satisfactory performance, during which a health professional would help the user interact with the PAW.

VII. Conclusion

In this paper, a novel PAW control design has been proposed for paraplegic wheelchair users. The assistive strategy is based on energy optimization, while maintaining a suitable fatigue level for users and using minimal electrical energy over a distance-to-go. This optimal control problem was solved by the online model-free reinforcement learning methods PoWER and GPOMDP. Their near-optimality was confirmed by the model-based approach finite-horizon Q-fuzzy iteration. An important contribution is that the near-optimality of finite-horizon Q-

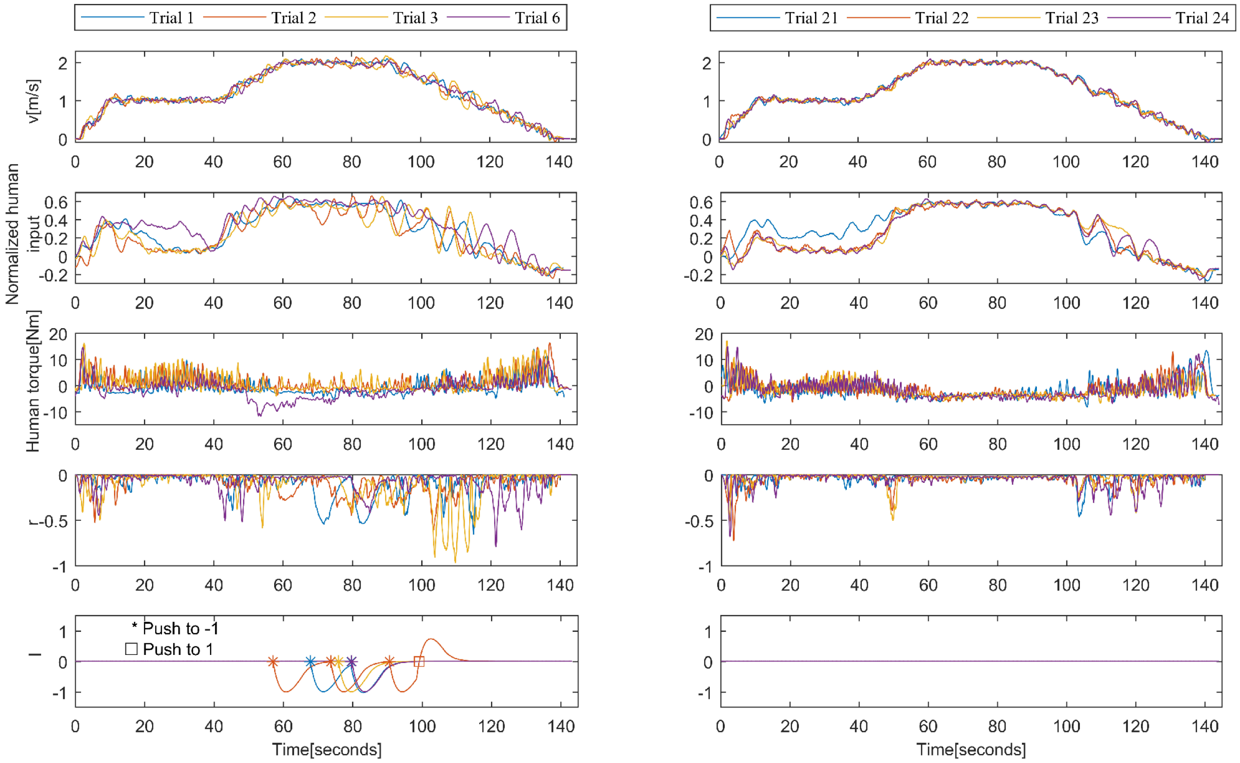


Figure 6: The trajectories of the first four stable trials and the last four trial. (The instant where the joystick is pushed is indicated on the I signal)

fuzzy iteration was proven. In addition, simulation results confirmed that PoWER with a simplified controller parameterization provides a considerably higher data efficiency, which renders the model-free framework better applicable in practice. Moreover, an investigation has been done to illustrate that PoWER is also able to adapt to human fatigue dynamics changes. Finally, a proof-of-concept experiment has been carried out to demonstrate the feasibility of the approach in practice. Future work will focus on validating the adaptability of the applied assistive algorithm for different users with the real wheelchair.

References

[1] World Health Organization. (2011). World report on disability.

[2] Guanetti, J., Formentin, S., Corno, M., & Savaresi, S. M. (2015, December). Optimal energy management in series hybrid electric bicycles. In *Annual Conference on Decision and Control (CDC),* (pp. 869-874). IEEE.

[3] Corno, M., Berretta, D., Spagnol, P., & Savaresi, S. M. (2016). Design, control, and validation of a charge-sustaining parallel hybrid bicycle. *IEEE Transactions on Control Systems Technology*, *24*(3), 817-829.

[4] Wan, N., Fayazi, S. A., Saeidi, H., & Vahidi, A. (2014, June). Optimal power management of an electric bicycle based on terrain preview and considering human fatigue dynamics. In *American Control Conference (ACC),* (pp. 3462-3467). IEEE.

[5] Seki, H., Ishihara, K., & Tadakuma, S. (2009). Novel regenerative braking control of electric power-assisted wheelchair for safety downhill road driving. *IEEE Transactions on Industrial Electronics*, *56*(5), 1393-1400.

[6] Tanohata, N., Murakami, H., & Seki, H. (2010, August). Battery friendly driving control of electric power-assisted wheelchair based on fuzzy algorithm. In *SICE Annual Conference* (pp. 1595-1598). IEEE.

[7] Mohammad, Sami, Thierry-marie Guerra, and Philippe Pudlo. "Method and device assisting with the electric propulsion of a rolling system, wheelchair kit comprising such a device and wheelchair equipped with such a device." U.S. Patent Application No. 15/310,073.

[8] Sutton, R. S., McAllester, D. A., Singh, S. P., & Mansour, Y. (2000). Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems* (pp. 1057-1063).

[9] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, *8*(3-4), 229-256.

[10] Buşoniu, L., Ernst, D., De Schutter, B., & Babuška, R. (2010, June). Online least-squares policy iteration for reinforcement learning control. In *American Control Conference (ACC),* (pp. 486-491). IEEE.

[11] Boyan, J. A. (2002). Technical update: Least-squares temporal difference learning. *Machine learning*, *49*(2-3), 233-246.

[12] Bellman, R., 1966. Dynamic programming. *Science*, *153*(3731), pp.34-37.

[13] Kober, J., & Peters, J. R. (2009). Policy search for motor primitives in robotics. In *Advances in neural information processing systems* (pp. 849-856).

[14] Fayazi, S. A., Wan, N., Lucich, S., Vahidi, A., & Mocko, G. (2013, June). Optimal pacing in a cycling time-trial considering cyclist's fatigue dynamics. In *American Control Conference (ACC),* (pp. 6442-6447). IEEE.

[15] Ronchi, Enrico, Paul A. Reneke, and Richard D. Peacock. "A conceptual fatigue-motivation model to represent pedestrian movement during stair evacuation." *Applied Mathematical Modelling* 40.7 (2016): 4380-4396.

[16] Tashiro, S., & Murakami, T. (2008). Step passage control of a power-assisted wheelchair for a caregiver. *IEEE Transactions on Industrial Electronics*, *55*(4), 1715-1721.

[17] Feng, G., Busoniu, L., Guerra, T. M., & Mohammad, S. (2018) Reinforcement Learning for Energy Optimization Under Human Fatigue Constraints of Power-Assisted Wheelchairs. In *American Control Conference (ACC),* (pp. 4117-4122). IEEE.

[18] Buşoniu, L., Ernst, D., De Schutter, B., & Babuška, R. (2010). Approximate dynamic programming with a fuzzy parameterization. *Automatica*, *46*(5), 804-814.

[19] Baxter, J., & Bartlett, P. L. (2000). Direct gradient-based reinforcement learning. In *International Symposium on Circuits and Systems* (Vol. 3, pp. 271-274). IEEE.

[20] Peters, J., & Schaal, S. (2006, October). Policy gradient methods for robotics. In *International Conference on Intelligent Robots and Systems,* (pp. 2219-2225). IEEE.

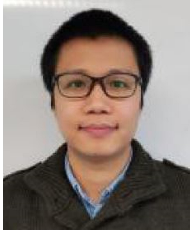
[21] Kober, J., & Peters, J. R. (2009). Policy search for motor primitives in robotics. In *Advances in neural information processing systems* (pp. 849-856).

[22] Deisenroth, M. P., Neumann, G., & Peters, J. (2013). A survey on policy search for robotics. *Foundations and Trends® in Robotics*, *2*(1–2), 1-142.

[23] Dayan, P., & Hinton, G. E. (1997). Using expectation-maximization for reinforcement learning. *Neural Computation*, *9*(2), 271-278.

[24] Ma, J., Zheng, Z., & Li, P. (2015). Adaptive dynamic surface control of a class of nonlinear systems with unknown direction control gains and input saturation. *IEEE Transactions on Cybernetics*, *45*(4), 728-741.

[25] Rodgers, M. M., Gayle, G. W., Figoni, S. F., Kobayashi, M., Lieh, J., & Glaser, R. M. (1994). Biomechanics of wheelchair propulsion during fatigue. *Archives of Physical Medicine and Rehabilitation*, *75*(1), 85-93.



Guoxi FENG received the M.Sc. degree in control engineering from the Université Polytechnique Hauts-de-France (UPHF), Valenciennes, France, in 2016. He is currently a PhD candidate at the UPHF. His current research interests include reinforcement learning and observer design for power-assisted wheelchair applications.

Lucian Buşoniu received the M.Sc. degree (valedictorian) from the Technical University of Cluj-Napoca, Romania, in 2003, and the Ph.D. degree (cum laude) from the Delft University of Technology, the Netherlands, in 2009. He is a Full Professor with the Department of Automation at the Technical University of Cluj-Napoca, where he leads the group on Robotics and Nonlinear Control. He has previously held research positions in the Netherlands and France. His research interests include nonlinear optimal control, reinforcement learning and approximate dynamic programming, multiagent systems, and robotics. He received the 2009 Andrew P. Sage Award for the best paper in the IEEE Transactions on Systems, Man, and Cybernetics.

Thierry-Marie Guerra received the Ph.D. degree in automatic control from the the Université Polytechnique Hauts-de-France (UPHF), France, in 1991 and the HDR degree in 1999. He is currently a Full Professor at the UPHF and Head of the CNRS Laboratory LAMIH http://www. univ-valenciennes.fr/LAMIH/. His current research interests include wine, hard rock, chess, nonlinear control, LPV, quasi-LPV (Takagi–Sugeno) models control and observation, nonquadratic Lyapunov functions and their applications to power train systems (IC engine, hybrid vehicles) and to disabled people. He is the Chair of IFAC T.C 3.2 “Computational Intelligence in Control”, member of the IFAC TC 7.1 Automotive Control, and Area Editor of the international journals Fuzzy Sets and Systems and IEEE Transactions on Vehicular Technology.

Sami Mohammad received the Ph. D. degree in automatic control from the university of Valenciennes and Hainaut-Cambrésis (UVHC), France, in 2011. He is now the CEO of Autonomad Mobility (www.autonomad-mobility.com), an innovative start-up specialized in high added-value mobility aids for disabled people. His research is focused on applied smart electrical mobility aids using advanced automatic control theories.