Optimal Control-On-Request: An Application in Real-Time Assistive Balance Control

Anastasia Mavrommati, Alex Ansari and Todd D. Murphey

***Abstract*— This paper presents a method for shared control where real-time bursts of optimal control assistance are applied by an observer on-demand to aid a simulated figure in maintaining balance. The proposed Assistive Controller (AC) calculates the optimal burst control fast, in real time, while accounting for nonlinearities of the dynamic model. The short duration of the AC signals allows a rapid transfer of control authority between the nominal and the assistive controller. This scheme avoids prolonged loss of nominal control authority on the part of the figure while facilitating the real-time integration of an external observer's guidance through the assistive control. We demonstrate the benefits of this control scheme in simulation using the Robot Operating System (ROS), in a context where the nominal controller fails to stabilize the figure and the AC is activated intermittently to not only keep it from falling but to additionally push it back to the upright position. The example signifies the efficiency of the proposed model-based AC even in the absence of force/pressure sensors. This approach presents an opportunity for using exoskeletons in balance support, fall prevention, and therapy. In particular, our simulation results indicate that a therapist equipped with an AC interface can, with minimal effort, increase active participation on the part of the patient while ensuring their safety.**

I. INTRODUCTION

A fundamental control problem for humanoid structures is balance maintenance and upright posture recovery. Commonly, simple models are employed to describe balance and derive control strategies [1], [2]. A big challenge in balance control lies in ensuring fall prevention regardless of the nature of the external disturbance. This issue has been addressed extensively but in most cases, sustained control paradigms have been proposed [3], [4]. This paper suggests that a fall may be sufficiently prevented if assistive optimal controls are calculated and applied in real time at the moment of fall detection based on sensor information about the current state of the system.

Moreover, the complexity of the problem is significantly magnified in the context of rehabilitative balance training. In the particular case of stroke rehabilitation, one of the main objectives is to improve posture stability [5]. There are essentially two different training methods to achieve this. First, during conventional non-automated balance therapy, the physical therapist manually assists patients, providing only as much assistance as needed to prevent a fall. However, continuous interaction with the therapist is both labor intensive and time consuming. In contrast, robot-assisted automated training promotes minimal interaction with the therapist, but tends to be ineffective as it fails to get the patient actively involved [6], [7]. This last issue is a consequence of the sustained control strategies commonly employed.

Considering the deficiencies of these two traditionally employed training methods, we propose an Assistive Controller (AC) that allows for an alternative robot-assisted training scheme where the therapist's guidance is integrated in real time with the assistive control, while making sure that the therapist's work load and interference remains low. To demonstrate the contributions of this assistive control approach, we consider the case where a standing stroke patient attempts to maintain an upright posture while in a robotic lower-limb exoskeleton. In this scenario, a monitoring physical therapist equipped with an AC interface (e.g. a simple button) could activate a corrective exoskeleton response on demand - i.e. as soon as the patient begins to fall. Upon activation of the AC, a burst of state-dependent optimal assistance is calculated in real time and applied for a very short duration, aiming not only to keep the subject from falling but to additionally push them back to the upright position. Immediately after, full control authority is ceded back to the patient (here acting as the Nominal Controller (NC)). This concept of shared control through intermittent bursts of optimal support comprises a unique assist-as-needed approach in that it completely avoids the issues of sustained control strategies where the interaction between two or more controllers acting on the same system simultaneously proves problematic. [1]

Standing balance aside, a similar problem of shared control is present in gait training robotic devices (i.e. lower-limb exoskeletons), where balance control is not automated and hence the use of supporting stability aids (e.g. crutches) is required (e.g. Ekso [10], Indego [11] etc.). Even when assist-as-needed training paradigms are employed, loss of balance is an issue and a safety harness is essential [12], [13], [14], [15], [16]. In this case of balance support during gait training, the AC can be used in the following context: with the NC now comprising both the embedded gait controller and the human effort, the therapist/observer is able to activate the AC when loss of balance is at stake. In this case, the purpose of the AC is to act momentarily in conjunction with the main embedded controller but without interfering with its

Authors are with the Neuroscience and Robotics Laboratory (N×R) at the Department of Mechanical Engineering (Mavrommati, Ansari, Murphey) and the Department of Physical Therapy and Human Movement Sciences (Murphey), Northwestern University, 2145 Sheridan Road Evanston, IL 60208, USA Email: stacymav@u.northwestern.edu; alexanderansari2011@u.northwestern.edu; t-murphey@northwestern.edu

[1] Interestingly, this scheme of intermittent instead of sustained control has been suggested to be natively used by the central nervous system (CNS) in human posture control in [8], [9].

performance in order to prevent a fall.

In accordance with the aforementioned scenarios, the proposed AC is designed in a way that facilitates the rapid exchange between human and computer control and provides quick automated assistance on demand. Upon request for assistance, the AC is able to assist the system as best as possible in the limited time frame available. Furthermore, the short duration of its activation ensures the feasibility of our design in a shared control setting. The mathematical foundation of the AC design was derived in [17] using optimal control principles of continuous and switched systems [18], [19], [20], [21], [22], [23]. Based on this, the AC can calculate, in real time, the optimal short-term controls using full state feedback. Unlike for LQR, the calculated controls account for the nonlinearities in the dynamic model.

In this paper, we test the real-time performance of the Assistive Controller acting on a simulated standing figure, using the Robot Operating System (ROS), a standard platform for real robotic applications [24]. The AC is shown to be suited for implementation on an embedded platform. In particular, our example verifies that real-time AC activation is feasible and requires only modest computational resources. In addition, it demonstrates how the proposed model-based AC is effective even in the absence of force/pressure sensors which are usually required for assist-as-needed techniques [15], [14].

This paper is structured as follows: Section II describes the concept of the AC design and reviews its mathematical foundation. The application in balance control is introduced in Section III, where the simulation platform is presented in detail and three example cases serve to illuminate the AC benefits in robot-assisted balance training. Finally, conclusion remarks are given in Section IV.

II. Assistive Controller Design

Authors in [17] derive a means to compute short burst optimal actions that improve the performance of nonlinear systems according to a general objective. The paper provides a simple linear system example to illustrate the concepts and demonstrate how optimal actions can help drive an unstable system toward the origin. In the following section, we provide an overview of the control synthesis methods in [17] and demonstrate how these can be adapted to a (nonlinear) shared control setting including the existence of a nominal controller representing the response of a human central nervous system (CNS).

A block diagram showing the functionality of the proposed controller in a shared control scenario is in Fig. 1. The system under control (e.g. a standing humanoid structure) is assumed to follow dynamics of the form:

$$\dot{x}(t) = f(x(t), u(t)) \tag{1}$$

where

$$f(x(t), u(t)) = g(x(t)) + h(x(t))u(t), \tag{2}$$

$x(t) \in \mathbb{R}^n$ is the state vector and $u(t) \in \mathbb{R}^m$ the input vector. The system representation is cast in a control-affine form, that allows non-linearity with respect to the state but assumes linearity in the control.

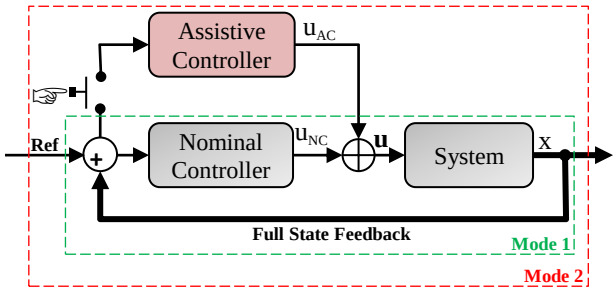

Fig. 1: Block diagram of burst assistive control. The NC and AC signals are only integrated momentarily when the button is pushed.

The switch in Fig. 1 is normally open (N.O.), so that when assistance is not requested, the system only receives input from the NC, i.e. $u = u_{NC}$, where u_{NC} is the input provided by the NC. From now on, this nominal operation is referred to as Mode 1 and its dynamics are:

$$\text{Mode 1:} \quad \dot{x}(t) = f_1(x(t), u_{NC}(t)). \tag{3}$$

To request assistance (e.g. when the patient starts falling), the observer (e.g. the physical therapist) pushes the button and the switch closes momentarily. For this short duration, the dynamics switch to the assistive mode (from now on referred to as Mode 2), where both the NC and AC control the system i.e. $u = u_{NC} + u_{AC}$, where u_{AC} is the input generated by the AC. The dynamics of this mode are:

$$\text{Mode 2:} \quad \dot{x}(t) = f_2(x(t), u_{NC}(t) + u_{AC}(t)). \tag{4}$$

After this short assistive control phase, the dynamics switch back to Mode 1 (3) and no further assistance is provided. With u_{NC} known, our purpose is to compute u_{AC} in an optimal way in order to get the best possible assistance in the limited time frame available. To measure the system's performance with or without assistance, we formulate a state tracking cost function, denoted as J_1:

$$J_1 = \int_0^T l_1(x(t))\, dt \tag{5}$$

where T is the final time of the system simulation. In a rehabilitation setting, J_1 might provide a measure for the weighted error from the balanced position.

Now, the objective of the AC is to calculate a control input u_{AC} so that the momentary switch to Mode 2 leads to desired change (typically reduction) in the tracking cost while conserving control effort. By change, we refer to a direct comparison between what the "nominal tracking cost" would be after nominal operation (3) and what the "new tracking cost" is expected to be after assisted operation (4). Even though the switch occurs for a finite duration, we may assume an infinitesimal duration $\lambda^+ \to 0$ and compute the change in cost $\frac{dJ_1}{d\lambda}$ as described in [17]. Specifically, it is

$$\frac{dJ_1}{d\lambda^+} = \rho(t)^T (f_2 - f_1) \tag{6}$$

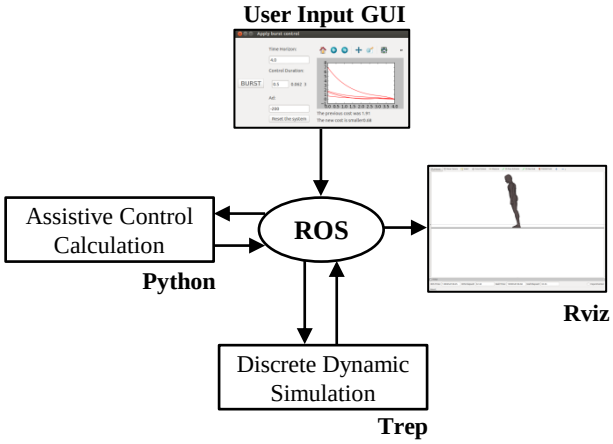

Fig. 2: Simulation setup. We use the Robot Operating System (ROS) to couple the processes in real time.

where $\rho(t) \in \mathbb{R}^n$ is the co-state variable calculated from the system trajectory during nominal operation based on the differential equation,

$$\begin{aligned} &\dot{\rho}(t) = -\mathcal{D}_x l_1(x(t))^T - \mathcal{D}_x f_1^T \rho(t) \quad \forall t \in [0, T] \\ &\text{subject to: } \rho(T) = \mathbf{0}. \end{aligned} \tag{7}$$

The time-varying scalar quantity in (6) (referred to in literature as the *mode insertion gradient* [25]) locally models the change in tracking cost for short durations near $\lambda \to 0^+$. Using (6), we construct a control objective J_2, as

$$\begin{aligned} J_2 &= \int_0^T l_2(x(t), u(t), \rho(t))\, dt \\ &= \frac{1}{2} \int_0^T [\frac{dJ_1}{d\lambda^+} - \alpha_d]^2 + \|u_{AC}(t)\|_R^2\, dt \end{aligned} \tag{8}$$

where $\alpha_d \in \mathbb{R}$ is a user-defined parameter indicating the desired change in the tracking cost. Cost J_2 is designed to select for optimal assistive controls that conserve control effort in achieving a desired change in tracking cost (5). The parameter α_d is important in that it allows for control over how aggressive the AC will be. For cost reduction, α_d should be set to a negative value. Weight matrix $R > \mathbf{0} \in \mathbb{R}^{m \times m}$ encodes the cost of control relative to tracking α_d.

By minimizing J_2 subject to $u_{AC}(t)$, the objective of the AC, as described previously, is achieved. As [17] proves, the value of control law $u_{AC}(t)$ that optimizes (8) is given by the algebraic expression,

$$u_{AC}(t) = (h(x(t))^T \rho(t) \rho(t)^T h(x(t)) + R)^{-1} [h(x(t))^T \rho(t) \alpha_d]. \tag{9}$$

For more detail and a more complete mathematical derivation of (9), see [17].

III. Shared Balance Support

A. Problem Formulation

This paper designs an assistive controller and assesses performance in shared balance control of a simulated standing, humanoid figure allowed to move in the sagittal plane. Here, we use a double inverted pendulum to model the humanoid.

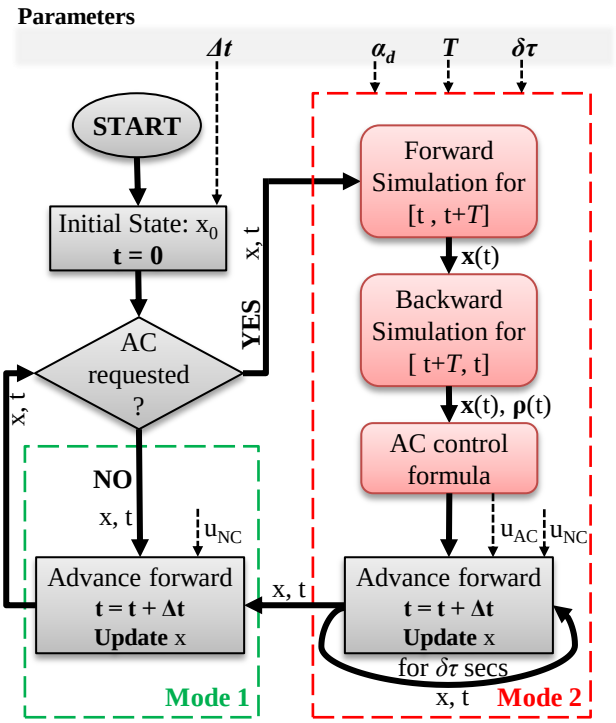

Fig. 3: Flow chart of a regular operation cycle as implemented in Python.

This is a common abstraction used extensively to examine the dynamics of humanoid or human balance [26], [27], [8], [3]. Alternative methods commonly control this system using simplified center of mass (COM) and center of pressure (COP) models [28], [4]. In this work we demonstrate results directly controlling the nonlinear double pendulum model for two main reasons. First, it demonstrates the scalability of the proposed AC approach. In particular, we show that the controller is fast and applies to nonlinear dynamics in a way that directly generalizes to models with different number of links and actuator placement.[2] Secondly, the choice to control to the double pendulum dynamics demonstrates flexibility. While the AC approach can be applied equally well to COP models, these models require specific force/pressure sensing hardware for feedback. Controlling to the double pendulum demonstrates how balance control can be achieved using alternative feedback based on joint angles and velocities.

The system model is illustrated in Fig. 4. Specifically, with both feet firmly planted to the ground and the knees locked in place, we allow 2 degrees of freedom, with the two pendulum pivot points corresponding to the hips and ankles respectively. The pendulum may be controlled through direct application of torque to the hip and ankle joints. Thus, the system has $n = 4$ states, i.e. the two joint angles $\boldsymbol{\theta}^{2\times1}$ and their velocities $\dot{\boldsymbol{\theta}}^{2\times1}$, and $m = 2$ inputs, i.e. the torques at the

[2]For example, we have applied assistive control to an underactuated 3 degree-of-freedom human model with actuated hip and knee joints and passive ankle joints.

joints. The mass, inertia and length of the model segments were determined from anthropometric data provided by [29]. The pendulum equations of motion take the form in (2) and can be derived using Lagrangian Mechanics and the Euler-Lagrange equation. The derivation is straightforward and will not be shown here.

Our performance objective in this application is to drive the system to the upright position, i.e. to keep the center of gravity (COG) at the center of the base of support. As there are no particular constraints on our choice for the tracking cost function, we employ a simple quadratic performance metric acting on the state. Referring back to (5), we select

$$l_1(x) = (x - x_{ref})^T Q(x - x_{ref}) \tag{10}$$

where $Q > \mathbf{0} \in \mathbb{R}^{n \times n}$ weights state errors relative to the balanced state. By conveniently using the vertical body position as an angle reference, our state reference becomes $x_{ref} = \mathbf{0}$.

For performance verification, we consider the example case where a human with impaired balance ability is in a standing position wearing a lower-limb robotic exoskeleton. In this scenario, the main system corresponds to the biomechanical model of a human augmented by a transparent overlying suit, while the NC plays the role of the CNS (central nervous system) attempting to maintain human balance. In literature, the CNS has been implemented both as a PID [26] and an optimal LQR controller [27]. Here, we chose to implement it as a PD controller providing feedback control on each of the joints. The AC is assumed to be embedded in the robotic suit. We emulate a scenario where the PD gains of the NC are insufficiently tuned and as a result, the provided torques fail to hold the figure upright.

B. Implementation in ROS

The overall simulation setup is illustrated in Fig. 2. The Robot Operating System (ROS), available online [24], lies in the center, coordinating the real-time communication between the running processes. ROS is a standard operating system widely employed for writing robot software. Here, we take advantage of its efficiency in coupling individually designed executables at runtime [24].

The humanoid balance model provided by the double inverted pendulum described in the previous section is simulated using the open-source software package called `trep` [30], which provides a simple Python interface that allows the user to model mechanical systems by creating tree representations of their geometry [31]. `trep` simulates the system dynamics implementing a variational integrator, instead of employing continuous numerical integration techniques. This feature allows us to test the efficiency of the AC on a discrete-time basis as real conditions would entail.

In addition we coded a simple Graphical User Interface (GUI) in Python that allows the user to activate the AC in real time by pressing a button on the computer screen. The GUI also enables tuning of the AC parameters prior to its activation. More details on the parameters are provided later in the paper. Finally, the system evolving in real time is visualized on the screen using Rviz, a ROS visualization tool.

The flow of a regular operation cycle is described in Fig. 3. The system simulation initiates at time $t = 0$ given an initial state $(\boldsymbol{\theta_0}, \dot{\boldsymbol{\theta}}_\mathbf{0})$. Recall that the integration of the system dynamics advances in discrete time steps due to the variational integrator employed. At each time step, the program checks whether an activation of the AC has been requested and enters one the modes described next.

Mode 1: With no assistance requested, the process simulates the system forward according to the NC input, and outputs the next system configuration.

Mode 2: If assistance is requested, the assistive control calculation module is triggered. The module solves for the state and co-state trajectories and evaluates the control formula (9) to compute the AC output. It applies the control for duration $\delta\tau$, before switching back to the nominal mode. Since, in theory, the AC is activated momentarily, the control duration should be defined approximately in the range of 0.01-0.03 s.

The simulation runs indefinitely, allowing the user to activate the AC in real time as needed. Fig. 4 demonstrates a sequence of Rviz snapshots where the button is pushed three times sequentially and the figure is driven to the upright position.[3]

C. Simulation Results

We will now present three simulation examples that illuminate the benefits of the Assistive Controller in robot-assisted balance training therapy. As an informal measure of performance, we specify approximate angle safety constraints at $\theta = \pm 0.2$. It is assumed that if the angles advance beyond these values, the Center of Gravity leaves the base of support and a fall occurs. Furthermore, we quantify the magnitude of nominal control effort as the energy E_{NC} of the signals $u^1_{NC}(t)$ and $u^2_{NC}(t)$, one for each joint. Specifically,

$$E_{NC} = \int_0^{T_f} \|u_{NC}(t)\|^2 dt \tag{11}$$

where T_f is either the final time of simulation or the time when the safety constraints are violated and the figure falls.

The AC parameters were tuned as follows: the time horizon $T = 3.0s$, the control duration $\delta\tau = 0.03s$ and the desired tracking cost reduction $\alpha_d = -200$. The weight matrices in (8) and (10) were set as

$$Q = \begin{pmatrix} 0.8 & 0 & 0 & 0 \\ 0 & 0.8 & 0 & 0 \\ 0 & 0 & 1.0 & 0 \\ 0 & 0 & 0 & 1.0 \end{pmatrix} \quad \text{and} \quad R = \begin{pmatrix} 0.1 & 0 \\ 0 & 0.1 \end{pmatrix}. \tag{12}$$

The higher weights on the velocities ensure a more realizable AC performance where fast motions are avoided. The parameter α_d defines how aggressive the assistance should be, i.e. either a "gentle" or a "strong push" towards the

[3] As shown in [17], it is possible to provide parameters that guarantee local stability and convergence in the case when the Assistive Controller is continuously activated.

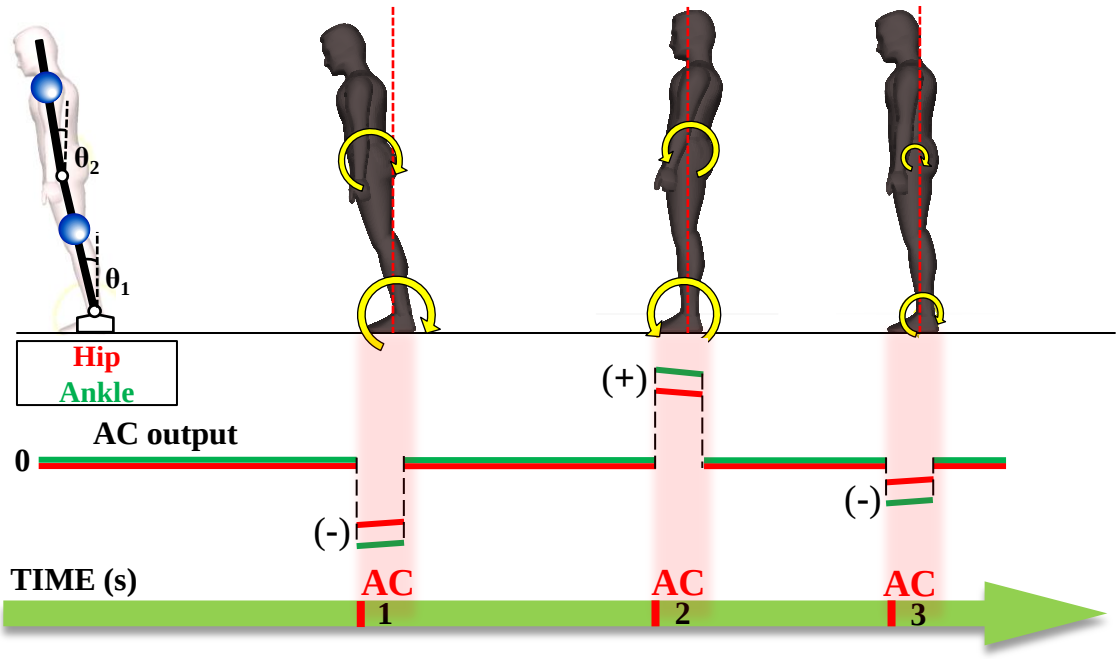

Fig. 4: Example of real-time standing balance control using the simulation setup. The AC is activated by pressing the button three times sequentially, eventually driving the system to the upright position.

balanced position. Therefore, by setting the absolute value of α_d relatively low, we allow for generation of more realizable and gentle controls for application to humans. Finally, by tuning the time horizon value T, we define how long the assistive control has to affect the change in cost. A value of $3.0s$ was selected, as it resulted in smooth angle trajectories (see Fig. 5).

In all cases described below, once the button was pressed, it took approximately 0.1s for the computation of the optimal assistive control in Python, on a laptop with an Intel Core i7 chipset. This confirms the feasibility of the on-demand AC activation in real time.

Case 1 - No assistance: Consider the case where a standing patient is allowed to balance without assistance i.e. only the NC is active (Fig. 5(a)). With no interference from the robot ($u_{AC} = 0$), the joint angles advance beyond the safety constraints indicating a fall. Therefore, although the training goal of active patient's involvement is achieved ($E^1_{NC} = 73.99$, $E^2_{NC} = 25.33$), there is *insufficient automated support* to prevent falls.

Case 2 - Burst assistance: In the second example, a monitoring physical therapist/observer is allowed to activate the AC, to provide on-line optimal assistance when deemed necessary. The effect of the AC activation is illustrated in Fig. 5(b). In the previous case the unassisted model violated the safety constraints, now the patient is safely "pushed" back towards the upright position and a potential fall is prevented. What's interesting is that the patient is still actively applying torques ($u_{NC} > 0$) and thus applies almost the same amount of control effort as in the first case ($E^1_{NC} = 80.31$, $E^2_{NC} = 18.72$). The robot interferes with the human effort only for a very short duration, before ceding full control authority back to the NC. Thus, in this case, both training goals are achieved: *the patient is actively involved and balance is maintained*. Notice that this concept is similar to the action taken by a conventional therapist while assisting a person to maintain balance by gently "pushing" them to the right direction as needed. However, here, almost no effort is required from the side of the therapist (automated assistance).

Case 3 - Intermittent assistance: Our third example considers the case where the AC is activated multiple times intermittently (Fig. 5(c)). It can be observed that not only does the state remain inside the safety constraints but also it successfully reaches the origin. In practice, this means that a fall has been prevented and also that the upright posture has been achieved. However, in comparison to the second case, this approach is not ideal because *the patient is essentially held in the upright position by the robot* without using their own power ($E^1_{NC} = 24.25$, $E^2_{NC} = 14.17$).

Therefore, the second case is the most effective application of the AC interface and indicates how an experienced therapist should take advantage of this control scheme to get the best therapy outcomes out of automated balance training.

IV. Conclusions and Future Work

This paper focuses on a shared control example where bursts of optimal assistance are applied to a simulated standing figure to prevent it from falling. This approach presents an opportunity for rehabilitation. In particular, robot-assisted balance therapy is challenging as control authority

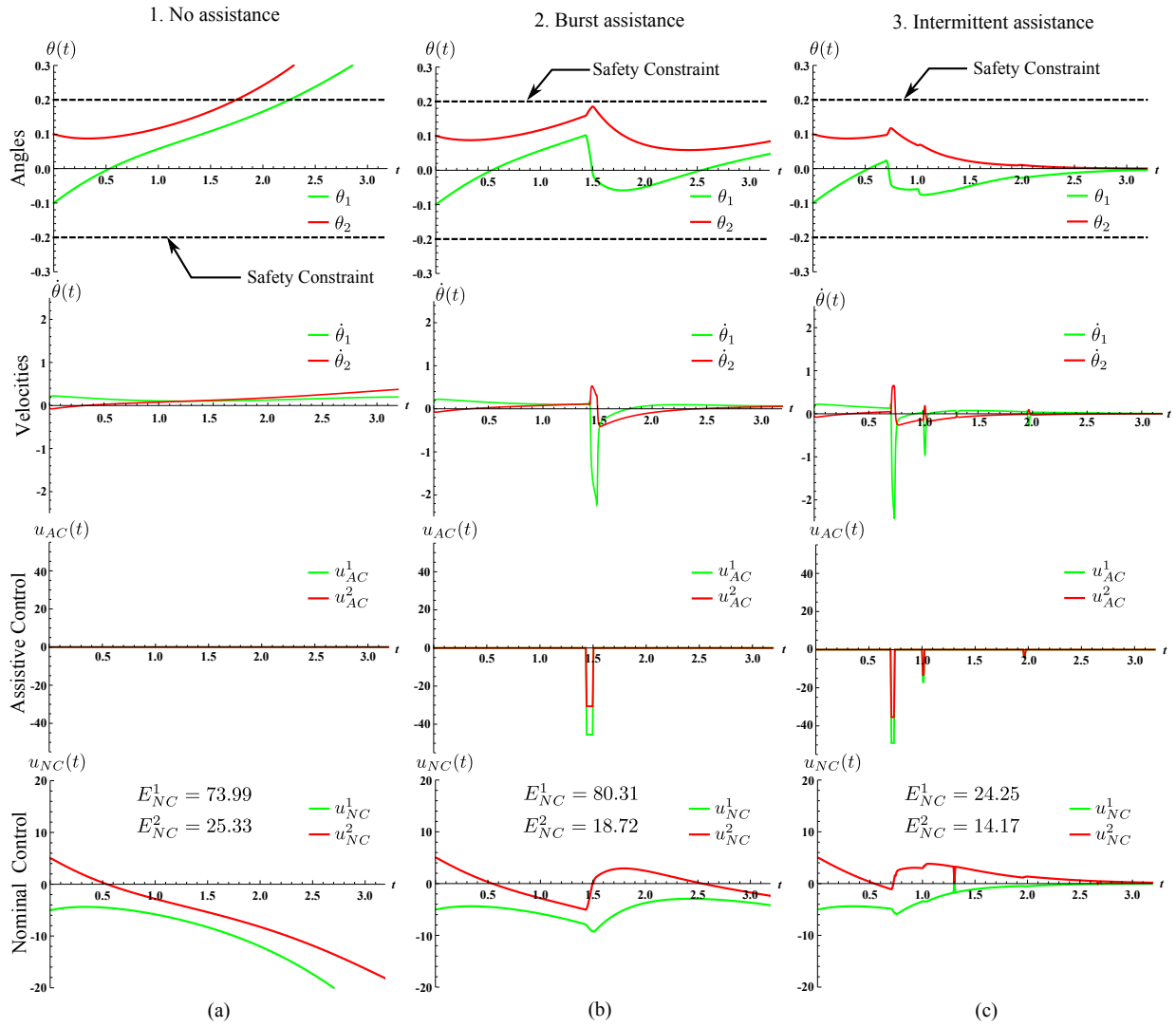

Fig. 5: Three examples demonstrate the AC benefits in robot-assisted balance therapy. u_{AC} indicates robot-applied torques and u_{NC} corresponds to human-applied torques. E_{NC} indicates total nominal control effort. (a) With no assistance, safety constraints are violated. (b) With burst assistance, balance is maintained with high energy contribution from the NC (preferred implementation). (c) With intermittent assistance, balance is ensured with low energy contribution from the NC.

is shared between the robot and the user, rendering their integration problematic. In an ideal rehabilitation setting, the robot should only provide as much assistance as needed to prevent a fall, requiring active patient participation. However, existing assistive control strategies either rely on sustained control or assume the use of force/pressure sensors which are generally not available in current over-ground lower-limb exoskeletons.

To address these issues, we proposed the application of an Assistive Controller (AC) that is activated on-demand and applies bursts of optimal control in real-time to prevent a fall. The short duration of activation ensures the feasibility of our design in a shared control scenario, alleviating the need for force feedback. The controller is *easy to implement*, has *low computational demands* and *runs in real time*. To demonstrate the aforementioned benefits, we implemented a simulated Assistive Controller based on ROS [24], with the model dynamics simulated on `trep` [30]. We showed that a therapist equipped with an AC interface can *ensure patient safety* while requiring *active participation*.

Lastly, as `trep` has been shown to be particularly suited for implementation on embedded platforms [32], future work will focus on embedding and testing the AC module in an experimental setting.

V. Acknowledgments

This material is based upon work supported by the National Science Foundation under awards CMMI-1200321 and IIS-1426961. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the

author(s) and do not necessarily reflect the views of the National Science Foundation.

References

[1] C. Golliday Jr and H. Hemami, "Postural stability of the two-degree-of-freedom biped by general linear feedback," *IEEE Trans. Autom. Control*, vol. 21, no. 1, pp. 74–79, 1976.

[2] H. Hemami and B.-R. Chen, "Stability analysis and input design of a two-link planar biped," *Int. J. Robot. Res.*, vol. 3, no. 2, pp. 93–100, 1984.

[3] B. Stephens, "Integral control of humanoid balance," in *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems , pages=4020–4027, year=2007*.

[4] M. Abdallah and A. Goswami, "A biomechanically motivated two-phase strategy for biped upright balance control," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 1996–2001.

[5] C. Winstein, E. Gardner, D. McNeal, P. Barto, and D. Nicholson, "Standing balance training: Effect on balance and locomotion in hemiparetic adults." *Arch. Phys. Med. Rehabil.*, vol. 70, no. 10, pp. 755–762, 1989.

[6] A. Wing, S. Allison, and J. Jenner, "Retaining and retraining balance after stroke," *Bailliere Clin. Neur.*, vol. 2, no. 1, pp. 87–120, 1993.

[7] D. S. Nichols, "Balance retraining after stroke using force platform biofeedback," *Phys. Ther.*, vol. 77, no. 5, pp. 553–558, 1997.

[8] I. D. Loram and M. Lakie, "Human balancing of an inverted pendulum: Position control by small, ballistic-like, throw and catch movements," *J. Physiol.*, vol. 540, no. 3, pp. 1111–1124, 2002.

[9] I. D. Loram, H. Gollee, M. Lakie, and P. J. Gawthrop, "Human control of an inverted pendulum: Is continuous control necessary? Is intermittent control effective? Is intermittent control physiological?" *J. Physiol.*, vol. 589, no. 2, pp. 307–324, 2011.

[10] H. Kazerooni, J.-L. Racine, L. Huang, and R. Steger, "On the control of the Berkeley lower extremity exoskeleton (BLEEX)," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2005, pp. 4353–4360.

[11] H. A. Quintero, R. J. Farris, and M. Goldfarb, "Control and implementation of a powered lower limb orthosis to aid walking in paraplegic individuals," in *Proc. IEEE Int. Conf. on Rehabilitation Robotics*, 2011, pp. 1–6.

[12] J. L. Emken, J. E. Bobrow, and D. J. Reinkensmeyer, "Robotic movement training as an optimization problem: Designing a controller that assists only as needed," in *Proc. IEEE Int. Conf. on Rehabilitation Robotics*, 2005, pp. 307–312.

[13] L. L. Cai, A. J. Fong, Y. Liang, J. Burdick, and V. R. Edgerton, "Assist-as-needed training paradigms for robotic rehabilitation of spinal cord injuries," in *Proc. IEEE Int. Conf. on Robotics and Automation*, 2006, pp. 3504–3511.

[14] S. K. Banala, S. H. Kim, S. K. Agrawal, and J. P. Scholz, "Robot assisted gait training with active leg exoskeleton (ALEX)," *IEEE Trans. Neural Syst. Rehabil. Eng.*, vol. 17, no. 1, pp. 2–8, 2009.

[15] S. Jezernik, G. Colombo, and M. Morari, "Automatic gait-pattern adaptation algorithms for rehabilitation with a 4-DOF robotic orthosis," *IEEE Trans. Robot. Autom.*, vol. 20, no. 3, pp. 574–582, 2004.

[16] M. Peshkin, D. A. Brown, J. J. Santos-Munné, A. Makhlin, E. Lewis, J. E. Colgate, J. Patton, and D. Schwandt, "Kineassist: A robotic overground gait and balance training device," in *Proc. IEEE Int. Conf. on Rehabilitation Robotics*, 2005, pp. 241–246.

[17] A. Ansari and T. Murphey, "Control-on-request: Short-burst assistive control for long time horizon improvement," in *Proc. American Control Conf.*, 2015.

[18] T. Caldwell and T. D. Murphey, "Single integration optimization of linear time-varying switched systems," *IEEE Trans. Autom. Control*, vol. 57, no. 6, pp. 1592–1597, 2012.

[19] T. D. Murphey, "On multiple model control for multiple contact systems," *Automatica*, vol. 44, pp. 451–458, 2008.

[20] E. Johnson and T. D. Murphey, "Second-order switching time optimization for nonlinear time-varying dynamic systems," *IEEE Trans. Autom. Control*, vol. 56, no. 8, pp. 1953–1957, 2011.

[21] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. S. Sastry, R. Bajcsy, and C. J. Tomlin, "A descent algorithm for the optimal control of constrained nonlinear switched dynamical systems," in *Proc. IEEE Int. Conf. on Hybrid Systems: Computation and Control*, 2010, pp. 51–60.

[22] Y. Wardi, M. Egerstedt, and M. Hale, "Switched-mode systems: Gradient-descent algorithms with armijo step sizes," *Discrete Event Dynamic Systems*, pp. 1–29, 2014.

[23] W. Zhang, A. Abate, J. Hu, and M. P. Vitus, "Exponential stabilization of discrete-time switched linear systems," *Automatica*, vol. 45, no. 11, pp. 2526–2536, 2009.

[24] (2014) Robot Operating System. Willow Garage. [Online]. Available: http://www.ros.org/

[25] M. Egerstedt, Y. Wardi, and H. Axelsson, "Optimal control of switching times in hybrid systems," in *Proc. IEEE Int. Conf. on Methods and Models in Automation and Robotics*, 2003.

[26] J. Milton, J. L. Cabrera, T. Ohira, S. Tajima, Y. Tonosaki, C. W. Eurich, and S. A. Campbell, "The time-delayed inverted pendulum: Implications for human balance control," *Chaos: Interdiscipl. J. Nonlin. Sci.*, vol. 19, no. 2, pp. 026 110–026 110, 2009.

[27] A. D. Kuo, "An optimal control model for analyzing human postural balance," *IEEE Trans. Bio-Med. Eng.*, vol. 42, no. 1, pp. 87–101, 1995.

[28] C. Ott, M. A. Roa, and G. Hirzinger, "Posture and balance control for biped robots based on contact force optimization," in *Proc. IEEE Int. Conf. on Humanoid Robots*, 2011, pp. 26–33.

[29] D. A. Winter, *Biomechanics and motor control of human movement*. John Wiley & Sons, 2009.

[30] (2014) `trep` software package development. [Online]. Available: trep.googlecode.com

[31] E. R. Johnson and T. D. Murphey, "Scalable variational integrators for constrained mechanical systems in generalized coordinates," *IEEE Trans. Robot.*, vol. 25, no. 6, pp. 1249–1261, 2009.

[32] J. Schultz and T. D. Murphey, "Embedded control synthesis using one-step methods in discrete mechanics," in *Proc. American Control Conf.*, 2013, pp. 5293–5298.