IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 2015.

Gradient-Driven Online Learning of Bipedal Push Recovery

Marcell Missura and Sven Behnke

***Abstract*— Bipedal walking is a complex and dynamic whole-body motion with balance constraints. Due to the inherently unstable inverted pendulum-like dynamics of walking, the design of robust walking controllers proved to be particularly challenging. While a controller could potentially be learned with a robot in the loop, the destructive nature of losing balance and the impracticality of a high number of repetitions render most existing learning methods unsuitable for an online learning setting with real hardware.**

We propose a model-driven learning method that enables a humanoid robot to quickly learn how to maintain its balance. We bootstrap the learning process with a central pattern generator for stepping motions that abstracts from the complexity of the walking motion and simplifies the problem setting to the learning of a small number of leg swing amplitude parameters. A simple physical model that represents the dominant dynamics of bipedal walking estimates an approximate gradient and suggests how to modify the swing amplitude to restore balance. In experiments with a real robot, we show that only a few failed steps are sufficient for our biped to learn strong push recovery skills in the sagittal direction.

I. Introduction

Robust walking is a key prerequisite for legged robots to unleash their full potential of traversing a variety of terrains. However, a reliable walk controller that enables two-legged robots to leave the safety of constrained laboratory environments has not yet been achieved. Leading methods, such as the one proposed by Kajita et al. [1], generate basic walking on flat surfaces, and reject only minor disturbances. Push recovery, walking on rough terrain, and agile control of the walking direction are topics of ongoing research.

Designing a robust walk controller is particularly challenging due to the fact that bipedal walking is a balance-critical motion. Steps must be landed in the right place at the right time, otherwise the walker may fall and sustain damage. The principal dynamics of a biped is similar to an inverted pendulum, which—once disturbed—quickly diverges away from the upright position. This property necessitates a timely response to disturbances in order to act before the state of balance escapes capturability. At the same time, walking is a rather complex motion due to the high number of degrees of freedom in the humanoid body. Real-time computation of a balance-constrained, high-dimensional walking motion requires abstraction using a low-dimensional model, which typically leads to modeling approximations or artificial constraints which prevent the exploitation of the energy-efficient natural dynamics of bipeds.

All authors are with the Autonomous Intelligent Systems Group, University of Bonn, Germany. Email: missura@ais.uni-bonn.de. This work has been partially supported by grant BE 2556/6-1 of German Research Foundation (DFG).

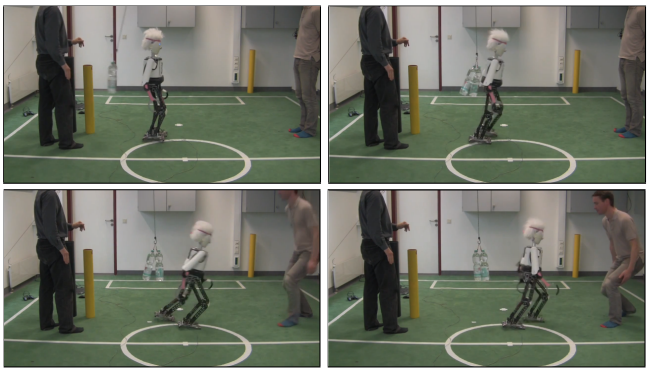


Fig. 1: Humanoid robot Copedo regains its balance after a strong push from the back.

Using machine learning with hardware in the loop is thought to be a promising alternative to analytic design for producing a capable controller. However, model-free learning of a high-dimensional motion generation for the motors of a humanoid robot such that a robust and controllable walk is achieved, is challenging. With the help of a motion skeleton and a fitness function that guides the learning process towards upright walking, classical evolutionary algorithms and reinforcement learning methods can be successful in simulation [2]. When a real robot is in the loop, the feasible number of trials and the risk of damaging the hardware become limiting factors, and dictate that the learning process must reach a reliable walking performance after a low number of experiences. Flight animals learn to walk within a few hours after birth, because their basic motion pattern generation and the dispostion to step in the right direction are innate and must only be tuned. This biological example suggests that initialization with a hard-wired motion pattern, and a rough concept of balance, is beneficial to successfully learn how to walk. We follow this paradigm and investigate a new learning method that incorporates a Central Pattern Generator (CPG), which produces coordinated stepping motions and simplifies the problem setting to the learning of leg swing amplitudes rather than whole-body control. A simple physical model speeds up the learning process by suggesting swing amplitude modifications with an estimated gradient. The learning algorithm is executed online during walking and improves the balancing capabilities of a biped by aggregating the feedback it gains from the physical model in a function approximator. We showcase our method in a real-robot experiment where a biped learns to recover its balance after being subjected to strong pushes from the back. Even though initially the learning controller has no notion of the right step size, a few steps are sufficient for it to learn

how to avoid a fall.

II. Related Work

Zero Moment Point (ZMP) preview control [1] is the most popular approach to bipedal walking. A number of pre-planned footsteps are used to define a reference trajectory for the ZMP—the base of the inverted pendulum under the foot. A continuous Center of Mass (CoM) trajectory that minimizes the ZMP tracking error is then generated by solving a quadratic programming problem in a Model Predictive Control setting [3]. The optimization is computationally expensive, but can be performed in real time. Whole-body control arises by computing joint trajectories with inverse kinematics in a way that the pelvis follows the CoM trajectory and the swing foot follows a smooth trajectory that connects the footstep locations in Cartesian space. The joint trajectories are tracked with high-precision position controlled motors. By using the ZMP preview control scheme, high-quality hardware [4] can walk reliably on flat ground as long as disturbances are small. Adaptive foot placement has only recently been achieved by Urata et al. [5], who replaced the computationally expensive CoM trajectory optimization with a faster method that samples a whole set of ZMP/CoM trajectory pairs. Triggered by a disturbance, their algorithm selects the best available motion according to given optimization criteria. Highly specialized hardware was used to carry out the real robot experiments. A drawback of the ZMP preview control algorithm is that the robot is forced to follow the motion of a low-dimensional model as closely as possible. This typically results in an unnatural, plane-restricted motion of the pelvis with bent knees. To preserve the theoretical stability of the precomputed CoM trajectory, precise position tracking requirements are imposed.

An entirely different approach to bipedal walking originates from passive dynamic walking pioneered by McGeer [6]. His experiments proved that not only control, but also actuation can be entirely removed from the system. The passive dynamics of legs with unactuated joints is sufficient to walk down a shallow slope. The graceful and energy-efficient motion of his bipedal constructions strongly resembles the human walk and suggests that the core principle of biological gaits may also be passive dynamics without control effort. Central pattern generated walking adds powerful actuation and control of the walking speed and direction, but no control of balance. In recent own work, we developed the Capture Step Framework [7] that complements a CPG gait [8] with balance control. The Capture Step Framework includes an analytic balance controller that uses the CPG to control the robot to step to the right locations in order to maintain its balance and track a commanded step size. It has been demonstrated to generate a stable, omnidirectional walk with strong disturbance rejection capabilities on a real robot[1] [9]. The non-intrusive balance control augmentation does not derogate the natural dynamics and allows the CPG to produce a high-dimensional walking motion with stretched knees.

[1]Video: `http://youtu.be/PoTBWV1mOlY`

As an alternative to restrictive analytic approaches, online learning strategies can potentially learn how to augment a gait with balance control. Rebula et al. [10], improved the reactive step of a simulated biped from a standing position by learning to step onto an offset from the capture point. Focusing on the walking speed, bipedal and quadrupedal gaits were successfully optimized using Policy Gradient methods in high-dimensional state spaces [11], [12]. With the same learning method, adjusted to a neuronal gait controller, the sagittal-only robot Runbot learned to walk fast and to cope with irregular terrain [13]. All these experiments started from an open-loop stable, hand-designed gait.

A quickly learning bipedal system was presented by Tedrake et al. [14]. Using online stochastic policy gradient estimation, the robot Toddler learns to walk on different surfaces in less than 20 minutes. The robot was designed in such way that it can passively walk down a slope without actuation. The success of this experiment can mostly be attributed to a strong simplification of the learning task to low-dimensional control of ankle actuation that imitates a passive dynamic gait without the need for a slope.

Yi et. al. [15] investigated online learning on real hardware using a reinforcement learning method. Their approach is also built on top of an open-loop gait trajectory generator and learns to optimize the input parameters of three biologically inspired disturbance rejection strategies. To make online learning on real hardware feasible, the reinforcement learning was simplified by a discretization of the input space and the assumption that the control parameters are restricted to parametric functions.

Morimoto et. al used Gaussian processes [16], [17] and receptive field weighted regression [18] to learn a Poincaré map that approximates the periodic dynamics of a biped. Using this map to select suitable actions, a policy gradient-based reinforcement learning method was used to train bipedal gaits in simulation and on real robots. Upright walking with an unspecified walking velocity in the absence of disturbances was achieved.

In previous own work, we published a learning framework [19] that learns to control the leg swing amplitudes of a CPG in order to improve the walking performance of a biped. The learning control can replace or coexist with the analytic balance controller of the Capture Step Framework. We investigated the aspects of learning to balance, reference tracking, and the robustness of the learning method in simulation. In this contribution, we focus on the most challenging part: learning to balance on real hardware.

III. Online Learning Framework

The architecture of our learning controller is illustrated in Fig. 2. In the following sections, we briefly introduce the components of the framework. We then turn our attention to the concepts of the learning process.

A. Bipedal Robot

A bipedal robot is an essential part of the control loop. It receives joint target positions $\boldsymbol{q}$ from the control software and

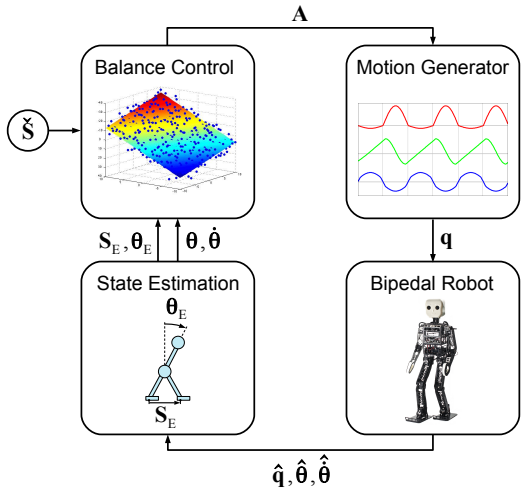


Fig. 2: The architecture of our online learning gait control framework. Using the sensor data received from the robot (bottom right), the State Estimation component (bottom left) determines the step size $\boldsymbol{S}_E$ and the trunk angle $\boldsymbol{\theta}_E$ at the end of each step. The Balance Control (top left) has the task of obeying the commanded step size $\check{\boldsymbol{S}}$ while maintaining the balance of the biped. From the errors the robot makes during walking, the Balance Control component learns the leg swing amplitude $\boldsymbol{A}$. The Motion Generator (top right) generates joint position targets $\boldsymbol{q}$ for periodic stepping motions with the desired leg swing amplitude $\boldsymbol{A}$.

provides sensor data about its internal state. It is equipped with joint encoders, accelerometers, and gyroscopes that measure the joint angles $\hat{\boldsymbol{q}}$ and allow for estimating trunk angle $\hat{\boldsymbol{\theta}}$ and angular velocity of the trunk $\hat{\dot{\boldsymbol{\theta}}}$, respectively. We have constructed a number of robots with the ability to tolerate the mechanical stress of a fall and to get back up on their feet. This skill is indispensable for learning how to walk, as falling is a natural part of the learning process. The humanoid robot Copedo we used to conduct our experiments is shown in Fig. 1. Copedo is 114 cm tall and weighs 8 kg. It is actuated by Robotis Dynamixel EX-106+ servos, which we operate in a low-gain position control mode in order to achieve compliant actuation. The emulated elasticity of the actuators results in a soft motion that forgivingly absorbs small disturbances and gracefully protects the motors from hardware damage and overheating, but it also increases the

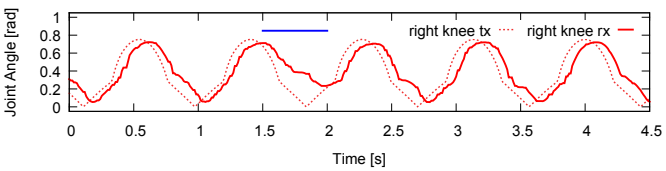


Fig. 3: Commanded (tx) and measured (rx) joint angles of the right knee during walking in place. To demonstrate the joint elasticity, the robot was pushed down on the right hip at 1.5 seconds. The blue bar indicates the time of the push. The position tracking error increases significantly in this phase.

position tracking error, making the robot more difficult to control. Fig. 3 shows joint angle data measured on the robot during a short period of in-place walking. During the experiment, the robot was pushed down on the right hip to demonstrate the compliant behavior. Online learning of motor control is an ideal way of combating imprecise actuation, latency, and sensor noise—challenges that are inherent to the operation of real hardware.

B. Motion Generator

To generate stepping motions, a low-level CPG [8] is used that produces periodic motion signals for the leg joints. It exposes a parameter vector $\boldsymbol{A} = (A_r, A_p, A_y)$ that contains dimensionless activation signals to control the leg swing amplitudes in the roll, pitch, and yaw directions. The sagittal, lateral, and rotational components of the physical step size $\boldsymbol{S}_E = (S_{Ex}, S_{Ey}, S_{E\psi})$ are unknown monotonous functions of the swing activation vector $\boldsymbol{A}$. The motion generator takes the responsibility of bounding and smoothing $\boldsymbol{A}$ in order to avoid hardware damage. The CPG combines well with our compliant actuator mode in a way that its configuration parameters allow to manually adjust the step motion pattern to account for the imprecision caused by the low-gain position control. The leg swing oscillation is generated by default with a nominal frequency. The step timing controller of the Capture Step Framework [7] adapts the flow of the motion phase in order to land the steps at the right time.

C. State Estimation

In the state estimation module, we reconstruct the tilted whole-body pose of the robot. We first estimate the trunk inclination $\boldsymbol{\theta} = (\theta_r, \theta_p)$ and angular velocity $\dot{\boldsymbol{\theta}} = (\dot{\theta}_r, \dot{\theta}_p)$ in the roll and pitch directions based on the accelerometer and gyroscope measurements. Then, we use the measured joint angles $\hat{\boldsymbol{q}}$ to set the kinematic model of the robot in pose using forward kinematics equations, and rotate the model around the center of the support foot such that the trunk angle matches the estimated values. When the swing foot has a lower vertical coordinate than the support foot, the roles of the feet are switched. In the moment of the support exchange, we obtain the step size estimate $\boldsymbol{S}_E$ by computing the distance and rotation between the feet, and the trunk angle $\boldsymbol{\theta}_E$. These two quantities are used to train the balance controller.

D. Balance Control

A higher control instance commands a step size $\check{\boldsymbol{S}}$ in order to steer the robot in its environment. This is illustrated on the top left of Fig. 2 as an input to the Balance Control module. The Balance Control has the task of obeying the commanded step size while maintaining the balance of the biped. During walking, the Balance Control component observes the state of balance and the error the robot makes with respect to the desired step size $\check{\boldsymbol{S}}$, and learns to improve the overall walking performance by controlling the step size using the leg swing activation parameter $\boldsymbol{A}$ of the CPG. In the next section, the learning process is introduced in more detail.

IV. Learning Method

Formally, the Balance Control is a function

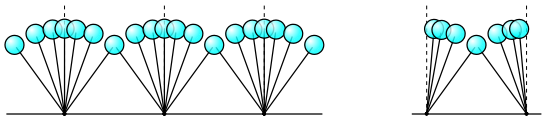
$$\boldsymbol{A} = \mathcal{B}(\boldsymbol{\theta}, \dot{\boldsymbol{\theta}}, \check{\boldsymbol{S}}), \tag{1}$$

which computes the leg swing amplitude activation vector $\boldsymbol{A} = (A_r, A_p, A_y)$ in response to the trunk angle $\boldsymbol{\theta} = (\theta_r, \theta_p)$, its angular velocity $\dot{\boldsymbol{\theta}} = (\dot{\theta}_r, \dot{\theta}_p)$, and the desired step size $\check{\boldsymbol{S}} = (\check{S}_x, \check{S}_y, \check{S}_\psi)$. While the complexity of function $\mathcal{B}$ is already much lower than what would be needed for whole-body control, we simplify it further by factorizing the Balance Control function into independent control functions

$$\begin{bmatrix} A_r \\ A_p \\ A_y \end{bmatrix} = \begin{bmatrix} \mathcal{B}_r(\theta_r, \dot{\theta}_r, \check{S}_y) \\ \mathcal{B}_p(\theta_p, \dot{\theta}_p, \check{S}_x) \\ \mathcal{B}_y(\check{S}_\psi) \end{bmatrix} \tag{2}$$

for the roll, pitch, and yaw leg swing activation signals. Please note that the roll leg swing activation corresponds to the lateral step size, the pitch leg swing activation corresponds to the sagittal step size, and the yaw leg swing activation corresponds to the rotational step size. The factorization is motivated by the dimensional decomposition of the walking motion illustrated in Fig. 4. Interestingly, the inverted pendulum-like motions in the sagittal and the lateral directions exhibit strongly distinct behaviors. In the sagittal plane, the center of mass crosses the pivot point of the pendulum in every gait cycle, while in the frontal plane, the center of mass oscillates between the support feet and never crosses the pivot point. We have used the dimensional decomposition concept in previous work to design an analytic balance controller [7]. The dimensional decomposition suggests that we can learn separate, possibly conceptually different controllers, for the sagittal and the lateral directions. We assume the effect of the rotational direction to be negligible for balance and model it with a simple mapping that converts the desired step size to the activation of the rotational leg swing amplitude. In this contribution, we concentrate entirely on the sagittal direction and show that due to the strong reduction of the input and output dimensions in combination with a model-based update rule, a robust sagittal balance controller can be learned from only a few steps.

We represent the sagittal Balance Control function $\mathcal{B}_p$ with a function approximator and initialize it with zero output. For the training of the function approximator, we measure two quantities of interest in the instant of the touch-down of the swing foot. We measure the state of the sagittal balance simply as the trunk pitch angle θ_{Ep} at the end of the step, and we measure the reference step size tracking error as $\bar{S}_x = S_{Ex} - \check{S}_x$. From these two quantities, we compute a gradient $\mathcal{G}(\theta_{Ep}, \bar{S}_x)$ to improve the physical step size for the conditions that were encountered during the step.



(a) Sagittal (b) Lateral

Fig. 4: Stick diagrams of a compass gait. (a) In the sagittal direction, the center of mass crosses the pendulum pivot point in every gait cycle. (b) In the lateral direction, the center of mass oscillates between the pivot points.

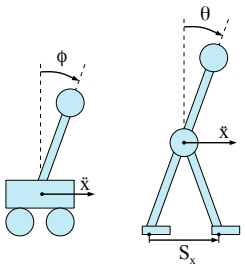


Fig. 5: The pendulum-cart model (left) resembles the angular dynamics of a biped (right). When the cart accelerates, the pendulum angle is accelerated in the negative direction. A biped accelerates by increasing its step size and can counteract undesired angular momentum.

To derive a suitable modification of the step size that attempts to maintain an upright posture of $\theta_p = 0$, we borrow a concept from the pendulum-cart model illustrated in Fig. 5. The simplest controller that manages to balance the pendulum on the cart when the pendulum angle $\phi \approx 0$, is a proportional controller $\ddot{x} = k\phi$ for a gain k. A biped is not a cart, but it can accelerate its center of mass by increasing or decreasing its step size. We can translate the proportional controller from cart to biped by converting acceleration to a step size modification and obtain a rough approximation of a balancing step controller

$$\Delta S_x \approx k\, \theta_{Ep}, \tag{3}$$

which estimates a sagittal step size modification ΔS_x to control the trunk pitch angle. For example, if at the end of a step the trunk pitch angle θ_{Ep} is positive, i. e. the robot is rotated "forward", the robot needs to take a larger step next time in the same situation in order to end up with a more upright posture.

As the right place to step to counteract undesired angular momentum does not necessarily coincide with the desired step size, a trade-off must be found between stepping into a desired location and avoiding a fall. We combine the control law (3) we derived from the cart-pendulum model, and the footstep error $\bar{S}_x$, into the gradient function

$$\mathcal{G}(\theta_{Ep}, \bar{S}_x) = \theta_{Ep} - p_\theta \tanh(p_S \bar{S}_x). \tag{4}$$

The characteristic saturation of the hyperbolic tangent function bounds the influence of the step size error $\bar{S}_x$ to a configurable limit of p_θ for two specific purposes: i) the parameterized saturation makes sure that the robot learns to track the reference step size carefully in order to avoid sudden changes of the CPG activation signal that are likely to cause instability, and ii) critical inclinations of $\|\theta_p\| \gg p_\theta$ dominate the gradient, ensuring balance takes priority over reference tracking when a fall is imminent. p_S is a weight

to fine-tune the influence of the step size error within the permitted bounds. Throughout our experiments, we used $p_\theta = 0.1$ and $p_S = 30$. Note that the gain k has been dropped from the gradient equation, because it is absorbed by the learning rate that we multiply the gradient with when we apply the update rule to the function approximator in Equation (5) below.

At the end of each step, we train the Balance Control function approximator with the update rule

$$\mathcal{B}_p(\theta_{p_i}, \dot{\theta}_{pi}, \check{S}_x) := \mathcal{B}_p(\theta_{p_i}, \dot{\theta}_{pi}, \check{S}_x) + \eta\, \mathcal{G}(\theta_{Ep}, \bar{S}_x), \forall i \in I, \tag{5}$$

where $\eta = 2.0$ is a learning rate, I is an index set, and $\{\theta_{p_i}, \dot{\theta}_{pi}\}, i \in I$ is the set of trunk pitch angles and angular velocities that were measured during the step. In words, we query the function approximator at the locations that were seen during the step, add the gradient to the returned values, and present the result as the new desired output to the function approximator.

The main control loop queries the Balance Control function approximator with a frequency of 83.3 Hz to drive the walking motion. It is essential that the function approximator can respond well within a 12 ms time frame, even when it is being updated with new data, in order to avoid blocking the main control loop. We use an open-source implementation of the LWPR [20] algorithm that fully satisfies our requirements.

V. Experimental Results

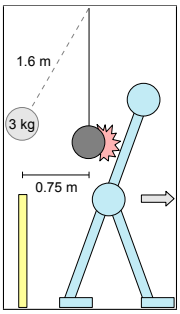


Fig. 6: Our experimental setup.

We demonstrate our method with an experiment performed with a real humanoid robot. The experimental setup is illustrated in Fig. 6. We swing a 3 kg mass attached to a 1.6 m rope onto the back of the robot from a distance of 0.75 m. The mass is pulled back and released by hand. A yellow pole marks the starting point to aid manual repeatability. The robot is positioned at the spot where the rope reaches the vertical position. We command the robot to walk in place by setting $\check{\boldsymbol{S}} = 0$. The Balance Controller is initialized with a zero step size and trained online during the experiment. The impact is rather strong and the Balance Controller has to learn to step forward in order to cope with the push, but as soon as balance is restored the robot should stop walking forwards and return to the commanded step size of zero. Although the experiment is in the sagittal direction, the motion of the robot is not restricted in any way. The step timing component of the Capture Step Framework is active. The step timing varies only slightly, but the use of the timing controller helps to avoid the accumulation of small errors that lead to lateral instability, which could interfere with our experiments. A video of this experiment[2] shows an uncut example of how Copedo successfully learns to absorb the impact. Due to the initialization with a zero step size and the strength of the impact, Copedo falls forward after the first push. However, the controller learns from the steps during the fall and is able to successfully absorb the second push. When observing the attempted steps following the first push in slow motion, one can see how the controller is learning and attempting to increase the step size. Photographs of the experiment are shown in Fig. 1. It is not necessary for the robot to fall in order to train the sagittal balance controller. Light pushes that drive the robot to the limit of its balance are sufficient. It highlights the robustness of the learning concept that it can learn even during a fall.

Fig. 7 shows the relevant data that allows us to analyze the learning process in detail. The experiment lasted 30 seconds in total. A first push was applied to the robot after 14 seconds, and a second push was applied after 24 seconds. The pushes are marked with vertical lines in the plots. The first and second plots show the trunk pitch and pitch rate, respectively. The trunk pitch values show how the robot fell forward after the first push, but not after the second push. In the pitch rate signal, two arrows mark the first peak after each push showing that both pushes were equally strong. The pitch rate signal is quite noisy. After the second push, the pitch rate is negative at 25 seconds, when the robot steps forward and corrects the trunk pitch angle to an upright position. The third plot shows the gradient that was computed after each step. The gradient increases and the robot learns during the three steps after the first push while it is falling forward and larger and larger steps would need to have been taken to regain balance. When the fall is detected at approximately 15.5 seconds, the gait controller switches to a fallen state, where walking and learning are suppressed. When an upright pose is detected, the controller automatically switches the walking and the learning back on. This is the case shortly before 20 seconds, where an undesired gradient can be seen to have been computed. Wrong gradients can be computed while the robot is standing, but is still in the process of manually being moved into the correct position. LWPR assigns more weight to new data and eventually forgets old data, and thus it makes sure that bad data do not degrade performance for an extended period of time. After the second push, the gradient stays near zero. The robot already learned

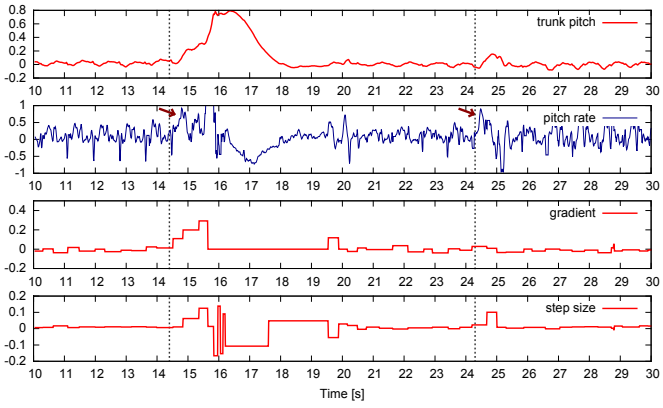


Fig. 7: Data recorded during a push recovery experiment.

[2]Video: http://youtu.be/qeWjy36gCBU

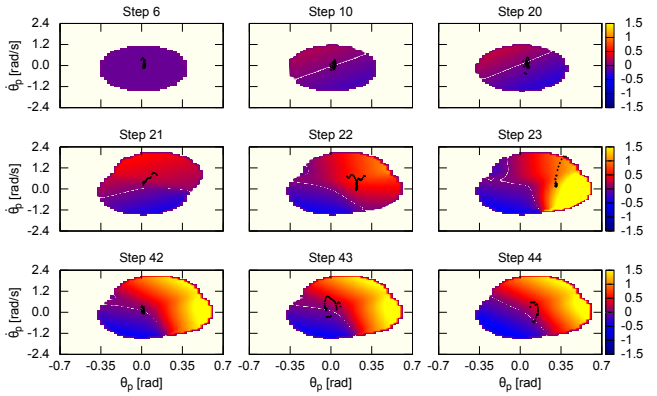


Fig. 8: Evolution of the learned feedback controller during the push-recovery experiment. The black dots mark the data points that were used to update the function approximator at the end of the step.

the correct step size to cope with the push and no further adjustment is needed. The fourth plot shows the step size measured during the experiment. It is interesting to note that the robot responded with a smaller step size to the second push than to the first push, and yet the robot did not fall. The evident reason for this is that since the controller already learned from the first push, it was able to react earlier to the second push and could balance the robot with less effort elegantly with a single step. As can be inferred from the inverted pendulum dynamics, the sooner the corrective action is taken, the smaller the correction needs to be.

The evolution of the function approximator during the learning experiment can be observed in Fig. 8. The black dots on top of the color maps of the function approximator show the data points that were used to update the controller after the last step. The first row consists of states of the function approximator before the first push. While the robot is walking in place, no significant change can be observed. We use a relatively large kernel size within the LWPR. The small cluster of data in the middle of the trunk pitch phase space is covered by a single linear kernel. The first push is followed by Steps 21, 22, and 23 in the second row, during which the robot is falling. The LWPR algorithm places new kernels along the observed trajectory of the pitch angle state in the phase space and covers a large portion of it with non-zero leg swing amplitude activation values. In the last row, Step 42 is the last step before the second push, where the robot is still stable. Step 43 is the capture step that absorbs the push, and, finally, Step 44 is a step of nearly zero size, during which the residual instability is dispersed passively.

VI. Conclusions

We presented a simple and robust online learning concept that achieves significant improvement of sagittal balance from a low number of experiences. The underlying strategy is to address the high-dimensional problem of walking motion generation by a Central Pattern Generator that encodes the coordination of single joints and exhibits a low-dimensional leg swing activation parameter that can be used to influence the step size. A dimensional decomposition of the balance control task into independent controllers for the lateral and the sagittal directions further simplifies the learning problem. Then, the sagittal leg swing is learned online based on error feedback and a simple pendulum-cart model assumption that allows the computation of an approximate gradient to correct the step size. A few steps are sufficient for the controller to produce convincing push-recovery capabilities. To our knowledge, the speed of our learning concept as well as the achieved sagittal balance are among the best results accomplished on a bipedal robot to date.

References

[1] S. Kajita, F. Kanehiro, K. Kaneko, K. Fujiwara, K. Harada, and K. Yokoi. Biped walking pattern generation by using preview control of zero-moment point. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2003.

[2] T. Geijtenbeek, M. van de Panne, and A. F. van der Stappen. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics*, 32(6), 2013.

[3] P.-B. Wieber. Trajectory free linear model predictive control for stable walking in the presence of strong perturbations. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2006.

[4] S. Kajita, M. Morisawa, K. Miura, S. Nakaoka, K. Harada, K. Kaneko, F. Kanehiro, and K. Yokoi. Biped walking stabilization based on linear inverted pendulum tracking. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2010.

[5] J. Urata, K. Nishiwaki, Y. Nakanishi, K. Okada, S. Kagami, and M. Inaba. Online decision of foot placement using singular LQ preview regulation. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2011.

[6] T. McGeer. Passive dynamic walking. *International Journal of Robotics Research*, 1990.

[7] M. Missura and S. Behnke. Omnidirectional capture steps for bipedal walking. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2013.

[8] M. Missura and S. Behnke. Self-stable omnidirectional walking with compliant joints. In *Workshop on Humanoid Soccer Robots*, 2013.

[9] M. Missura and S. Behnke. Balanced walking with capture steps. In *RoboCup 2014: Robot Soccer World Cup XVIII*. Springer, 2014.

[10] J. Rebula, F. Canas, J. Pratt, and A. Goswami. Learning capture points for humanoid push recovery. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2007.

[11] T. Hemker, M. Stelzer, O. von Stryk, and H. Sakamoto. Efficient walking speed optimization of a humanoid robot. *International Journal of Robotics Research*, 28(2):303–314, 2009.

[12] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2004.

[13] T. Geng, B. Porr, and F. Wörgötter. Fast biped walking with a sensor-driven neuronal controller and real-time online learning. *International Journal of Robotics Research*, 2006.

[14] R. Tedrake, T. W. Zhang, and H. S. Seung. Learning to walk in 20 minutes. In *14th Yale WS on Adaptive and Learning Systems*, 2005.

[15] S.-J. Yi, B.-T. Zhang, D. Hong, and D. D. Lee. Online learning of a full body push recovery controller for omnidirectional walking. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2011.

[16] J. Morimoto and C. G. Atkeson. Nonparametric representation of an approximated Poincaré map for learning biped locomotion. In *Autonomous Robots*. Springer, 2009.

[17] J. Morimoto, C. G. Atkeson, G. Endo, and G. Cheng. Improving humanoid locomotive performance with learnt approximated dynamics via guassian process for regression. In *IEEE Int. Conf. on Intelligent Robots and Systems (IROS)*, 2007.

[18] J. Morimoto and C. G. Atkeson. Learning biped locomotion. In *IEEE Robotics and Automation Magazine*, 2007.

[19] M. Missura and S. Behnke. Online learning of balanced foot placement for bipedal walking. In *IEEE-RAS Int. Conf. on Humanoid Robots (Humanoids)*, 2014.

[20] S. Vijayakumar, A. D'souza, and S. Schaal. Incremental online learning in high dimensions. *Neural Computation*, 17(12):2602–2634, 2005.