Efference Copies in Neural Control of Dynamic Biped Walking

P. Manoonpong [a], F. Wörgötter [a,*]

[a] *Bernstein Center for Computational Neuroscience (BCCN), University of Göttingen, D-37073 Göttingen, Germany*

Abstract

In the early 1950s, von Holst and Mittelstaedt proposed that motor commands copied within the central nervous system (efference copy) help to distinguish 'reafference' activity (afference activity due to self-generated motion) from 'exafference' activity (afference activity due to external stimulus). In addition, an efference copy can be also used to compare it with the actual sensory feedback in order to suppress self-generated sensations. Based on these biological findings, we conduct here two experimental studies on our biped "RunBot" where such principles together with neural forward models are applied to RunBot's dynamic locomotion control. The main purpose of this article is to present the modular design of RunBot's control architecture and discuss how the inherent dynamic properties of the different modules lead to the required signal processing. We believe that the experimental studies pursued here will sharpen our understanding of how the efference copies influence dynamic locomotion control to the benefit of modern neural control strategies in robots.

Key words: Legged robots; Recurrent neural network; Adaptive walking; Dynamic walking; Internal model

1 Introduction

Neural networks have become a versatile tool in many applications like pattern recognition, function approximation and others. Until recently networks, however, have not been used so often for the control of machinery (e.g., robots).

* Corresponding author. Tel.: +49 (0) 551 5176-528; fax: +49 (0) 551 5176-449.
Email addresses: poramate@nld.ds.mpg.de (P. Manoonpong), worgott@nld.ds.mpg.de (F. Wörgötter).

The difficulty in relaying network output to the end-effectors in a coordinated way and the complex structure of motor control networks may explain why they are still not much used for solving complex motor control problems so far. Recently, a few studies suggested, however, that small networks can be very powerful for addressing such problems. The works of Ijspeert et al. (2007) [13], Bem et al. (2003) [1], and Meyer et al. (2003) [28] have shown that in robots complex movement patterns like swimming and walking can be controlled and coordinated by neural network activity. The employed machines (lamprey or salamander like robots) are this way able to produce undulatory movements navigating through their environment. In our own studies, we have used an adaptive neural network to control a dynamic biped robot, called "RunBot" [8], [22]. This machine is able to walk and learn to adapt its posture and gait parameters to different terrains, e.g., when walking up a slope.

While this shows the power of network control, at least one important problem has so far not been addressed: All moving system are - on the sensor side - faced with noise. This could be random noise from various sources in the environment (external noise) but also disturbances which are introduced by the ego-motion (internal noise). For example, every step stimulates our own vestibular system in an unwanted way. Both noise sources mask other more relevant stimulus events and lead to reduced performance of the sensor system.

The brains of animals have developed strategies to compensate for these noise sources and the goal of this study is to show that these strategies can also be copied efficiently into robots allowing the machines to ignore external as well as internal noise. More than that: As internal noise is repetitive while walking it is predictable. This leads to the situation that the robot can recognize the disturbance. The comparison between the expected internal noise and the one actually measured can be used as an error signal which drives network learning as will be shown below. To this end we will employ the idea of "efferent copies".

Around the mid-19th century, von Holst and Mittelstaedt (1950) [10] demonstrated in animal models that motor commands are copied within the central nervous system (CNS). These copies help to distinguish 'reafference' (afference activity due to self-generated motion) from 'exafference' (afference activity due to changes in the external world). They can be also used for comparison with the actual sensory feedback in order to subtract self-generated sensations for maintaining stable perception. Similarly, Sperry (1950) [35] presented evidence which supported this idea. He showed that sensory areas receive discharge patterns (efference copy) with respect to the *expected* sensory feedback. In the early 1960s, Held (1961) [9] indicated that efference copies and the reafference generated by self-motion cannot be directly compared due to the different dimensionality between motor commands and sensory feedback. Therefore, he proposed a neural structure that transforms an efference copy signal into an expected sensory input to be able to compare it to the actual incoming sensory

signal. This neural transformation mechanism is known as "internal model" [40]. As described by Kawato (1999) [18], internal models or internal loops of biological systems are classified into three types: Inverse internal model (the system calculates a motor command from a desired trajectory/state information), forward internal model (the system predicts sensory consequences from efference copies), and integrated internal model (the system integrates both inverse and forward models).

Based on the biological findings described above, several robot experiments have been performed applying efference copy and internal model concepts employing these ideas for arm control [29], visuo-acoustic coordination [33] as well as leg control [5],[20] (see the Discussion section for details). These studies show that the efference copy principle together with an appropriate internal model can be successfully applied to a wide range of robot control problems. The work presented here extends this line of research to problems in dynamic walking control. In the present study we conduct two experiments on "RunBot": 1) The first experiment shows that an efference copy can be applied to eliminate external and self-generated sensory noise. Normally such perturbations destabilize the activation parameters for the gait and cause unstable walking. This can be successfully avoided by using an efference copy signal. 2) In the second experiment, we demonstrate that the robot can detect a slope by the deviation of its own gait from the normal gait-pattern observed on flat ground. This deviation signal can be used for learning the new parameter set, applicable to slope-walking.

The employed networks in general consists of three components: A network for basic walking, a learning control network as well as an efferent copy and internal model building network. In total this leads to a somewhat higher complexity of the network structure. As compared to the original control network of RunBot [8] the three modules, however, can be understood one by one, which makes network design simple. Thus, the main purpose of this article is not only to present the applications of the efference copy for dynamic locomotion control. In addition to this, some emphasis is put on the modular design and the aspect how the inherent dynamic properties of the different modules lead to the required signal processing.

In the following section, we give a general overview of the RunBot system. Afterwards mechanical setup of RunBot and adaptive reflex neural locomotion control forming the system's basic behavior are presented in brief where the complete descriptions can be found in our previous publications [8], [22]. Sections 3 and 4 show experimental studies for the application of the efference copy for improving locomotion control and determining terrain condition changes which is the main contribution of the article. Discussion and conclusions are provided in Sections 5.

2 RunBot system

The RunBot system (see Fig. 1) [22] uses the design principle of multiple nested loops to couple its mechanics with adaptive reflex neural locomotion control through an environment. Employing this hierarchical architecture, RunBot exhibits the self-stabilizing and passive properties [8] reflected by its mechanics. It can stably walk with different speeds regulated through its reflexive neural control [8]. Furthermore, it can adapt its gaits to different terrains by means of a neural learning process using adaptive neural control [22]. An overview of the mechanical setup and adaptive reflex neural locomotion control are provided in the following, for more details see [22].

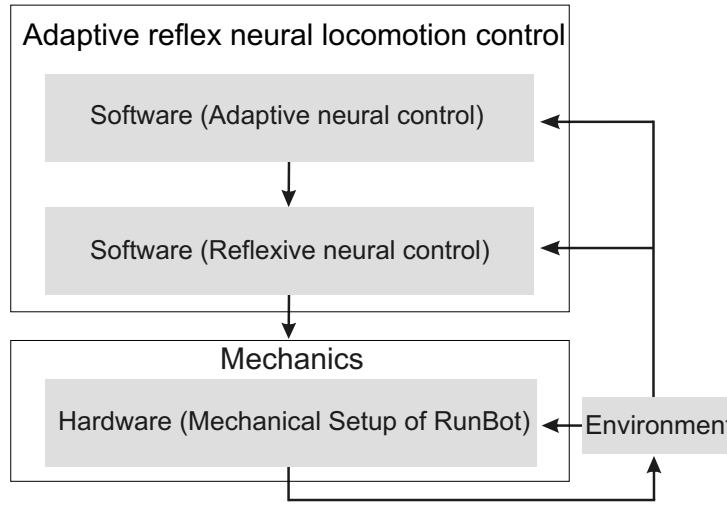

Fig. 1. The RunBot system. It is divided into three levels (Mechanics, Reflexive neural control, and Adaptive neural control) organized as a hierarchical structure and coupled via the environment.

2.1 Mechanical setup of RunBot

RunBot is a planar dynamic biped robot (see Fig. 2). It consists of four actuated joints: left hip, right hip, left knee and right knee. Each joint is driven by a modified servo motor where the built-in Pulse Width Modulation (PWM) control circuit is disconnected, while its built-in potentiometer is used to measure the joint angles (S). RunBot has no actuated ankle joints, resulting in very light feet and efficiency for fast walking. Its feet were designed having a small circular form (4.5 cm long). Each foot is equipped with a ground contact sensor (G). A mechanical stopper is implemented on each knee joint to prevent it from going into hyperextension. Approximately seventy percent of the robot's weight is concentrated on its trunk and the parts of the trunk are assembled in a way that its center of mass is located forward of the hip axis.

In addition, it has an upper body component (UBC), which can be actively moved to shift the center of mass backward or forward for walking on different terrains, e.g., level floor versus up or down a ramp. It leans backwards during walking on a level floor (see Fig. 2b) and this position is also suitable for walking down a ramp [23]. On the contrary, it will lean forwards (reflex action) when RunBot falls backwards or after it successfully learned to walk up a ramp (see Fig. 2b). The corresponding reflex is controlled by an accelerometer sensor (AS) functioning as its simple vestibular system. The AS is installed on top of the right hip joint. In addition, one infrared (IR) sensor is implemented at the front part of RunBot pointing downwards to detect a ramp. Here, the IR sensor serves as a simple vision system, which can distinguish between a level floor with black color and a painted ramp with white color (see Fig. 2b). This sensory signal is used for adaptive control. All sensory and motor signals are converted through a AD/DA converter board (USB-DUX [1]) with the update frequency of 250 Hz.

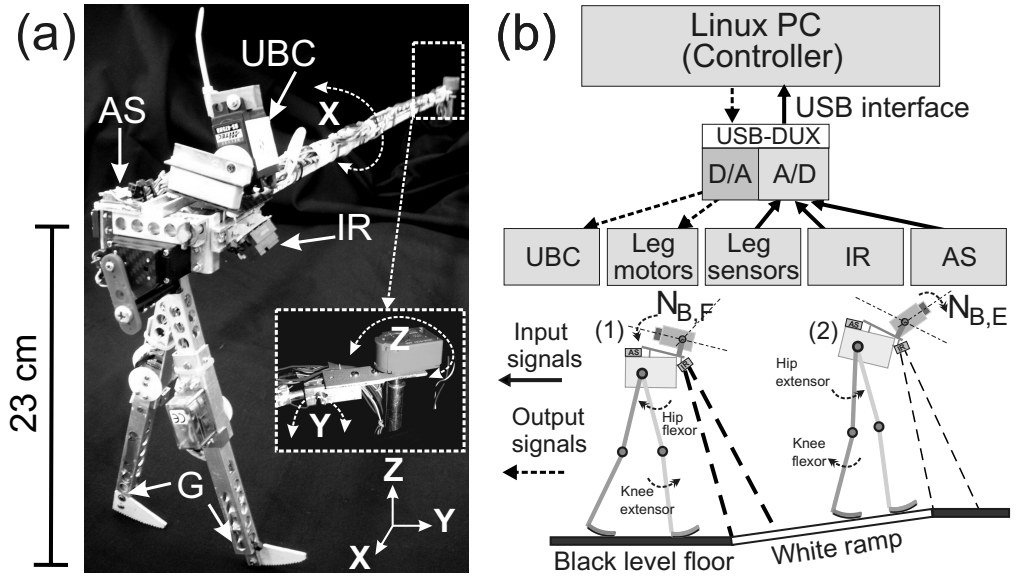

Fig. 2. (a) The planar dynamic robot RunBot. UBC, upper body component; IR, infrared sensor; AS, accelerometer sensor; G, ground contact sensor. (b) Schematic set-up of the RunBot system. Leg sensors consist of joint angle and ground contact switch sensors, leg motors are the motors of the left and right hip and knee joints. The detection range of the IR sensor for slope sensing is shown in the lower figure where the thick dashed ray of the IR sensor (1) indicates that the sensor gives a high output signal while the thin dashed ray (2) means a low signal. Hence the sensor responds more strongly to the white color. $N_{B,F}$, body flexor (leaning backwards); $N_{B,E}$, body extensor (leaning forwards).

We constrain RunBot in the sagittal plane by a boom of one meter length. RunBot is attached to the boom via a freely rotating joint in the x-axis, while

[1] http://www.linux-usb-daq.co.uk.

the boom is attached to the central column with freely rotating joints in the y and z axes (see Fig. 2a). The mechanical design of RunBot has the following special features that distinguish it from other powered biped robots and that facilitate high-speed walking and exploitation of natural dynamics: (a) small, curved feet allowing for rolling action; (b) unactuated, hence light, ankles; (c) lightweight structure; (d) light and fast motors; (e) proper mass distribution of the limbs; and (f) properly positioned mass center of the trunk. Utilizing all these properties, RunBot can perform self-stabilization of gaits and it also exhibits passive walking characteristics reflected by the fact that during one quarter of its step cycle all motor voltages remain zero [8].

2.2 Neural locomotion control

The neural locomotion control (see Fig. 1) consists of two main structures: the adaptive and reflexive neural control circuits. All neurons in the circuits are modeled as rate-coded neurons with the standard sigmoid transfer function. They are simulated on a Linux PC with an update frequency of 250 Hz.

2.2.1 Reflexive neural control

The reflexive neural control is based on several reflex mechanisms. It is composed of two submodules. One is for leg control and the other is for UBC control. Both leg and UBC controls are independent but they are indirectly coupled through the mechanics of RunBot (see Figs. 2 and 3).

The leg control, simulated as mono-synaptic connections, contains motor neurons (N), which are linear and can send their signals unmodified to the motors (M) (see Figs. 3 and 4). There are several local sensor neurons (proprioceptor), which, by their conjoint reflex-like actions, trigger different gaits, e.g., slow and fast. These local sensor neurons can be classified into three loops: joint control, intra-joint control and leg control. Joint control arises from angle sensors S at each joint (Local 1, see Fig. 3), which measure the joint angle and influence only their target motor neurons. Intra-joint control is achieved from sensors A (Local 2, see Fig. 3), which measure the anterior extreme angle (AEA) at the hip and trigger an extensor reflex at the corresponding knee. Leg control comes from ground contact sensors G (Local 3, see Fig. 3), which drive the motor neurons of all joints.

The UBC control represents a long-loop reflex, which is indirectly modulated by its AS through the adaptive neural control network (see Fig. 3). In general situations like when walking on flat terrain, the AS is inactive and the flexor body motor neuron $N_{B,F}$ is activated to lean the body backwards (see Fig. 2b) while the extensor motor neuron $N_{B,E}$ is inhibited. This situation is reverted

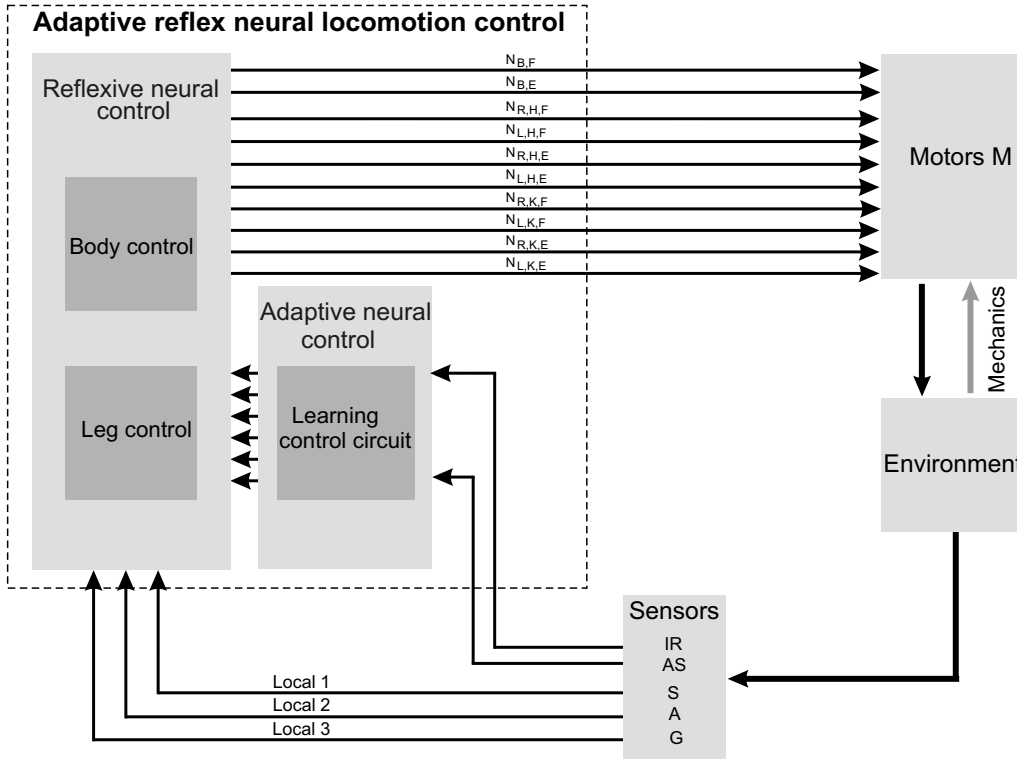

Fig. 3. Neural locomotion control (see text). Reflexive walking behavior arises from the interaction of three local sensorimotor loops (reflexive neural control) together with the passive properties (mechanics). Additionally, adaptation is achieved by a learning mechanism. A gray arrow represents RunBot's physical embodiment eliciting its passive dynamic walking properties. IR, infrared sensor; AS, accelerometer sensor; S, joint angle sensor of hips and knees; A, stretch receptor for anterior extreme angle (AEA) of the hips; G, ground contact sensor; $N_{B,F}$, flexor body-motor signal; $N_{B,E}$, extensor body-motor signal; $N_{R,H,F}$, flexor leg-motor signal of the right hip; $N_{L,H,F}$, flexor leg-motor signal of the left hip; $N_{R,H,E}$, extensor leg-motor signal of the right hip; $N_{L,H,E}$, extensor leg-motor signal of the left hip; $N_{R,K,F}$, flexor leg-motor signal of the right knee; $N_{L,K,F}$, flexor leg-motor signal of the left knee; $N_{R,K,E}$, extensor leg-motor signal of the right knee; $N_{L,K,E}$, extensor leg-motor signal of the left knee. In general, indices are omitted below the last relevant level, e.g., N_R applies to flexor and extensor of the hip and knee of the right leg.

when a strong signal from the AS exists, which happens only when RunBot falls backwards, e.g., RunBot tries to walk up a ramp. This will trigger a leaning reflex of the UBC. More detailed descriptions of all neuron models together with the neural network structures and the discussion of their parameters can be found in [22].

2.2.2 Adaptive neural control

RunBot's task was to learn walking up a ramp and then continue again on a level floor. The learning goal is to avoid the leaning reflex and thereby learn

to also change gait parameters in an appropriate way to prevent RunBot from falling. We use adaptive neural control to change the leaning action of the UBC by learning and to also influence several other leg control parameters for gait adaptation. This is accomplished by using six learner neurons changing activation parameters of their target neurons (see Fig. 4). Our learning algorithm (described in details later) applies a correlation based differential Hebbian learning rule [31] where the modification of all those parameters will be controlled by two kinds of input signals: one is an early input (called predictive signal) and the other is a later input (called reflex signal). In general, we use the IR signal as a predictive signal while the AS signal serves as a reflex signal (see Fig. 4). At the beginning, the connections between the predictive signal and learner neurons converge with zero strengths (dashed arrows in Fig. 4). In this situation, parameters of the target neurons will be altered only by the reflex signal (solid arrows between the reflex signal and learner neurons in Fig. 4); i.e., the leaning reflex of the UBC together with the gait adaptation will be triggered by the AS signal as soon as RunBot falls. Hence, RunBot will begin to walk up the ramp with a wrong set of gait parameters and an inappropriate posture of the UBC. Thus, it will eventually fall leading to a signal at the AS, which will change RunBot's parameters but too late (when it already lies on the ground). Due to learning the modifiable synapses (ρ_1^1,...,ρ_1^6, dashed arrows in Fig. 4), which connect the predictive IR signal with the learner neurons ($\mathrm{L}_1, ..., \mathrm{L}_6$), will grow. Consequently, after 3-5 falls during the learning phase, gait adaptation together with posture control of the UBC will finally be driven by the predictive IR signal instead. Correspondingly, RunBot will adapt its gait together with leaning the UBC in time. The used learning algorithm has the property that learning will stop when the reflex signal is zero [31]; i.e., when RunBot does not fall anymore. On returning to flat terrain, the IR output will get small again and RunBot will change its locomotion and posture back to normal for walking on a level floor.

Learning algorithm: In general, each learner neuron $\mathrm{L_n}$ requires two input signals (u_0, u_1) with synaptic weights (ρ_0, ρ_1) (see solid frame in Fig. 4). Here, we use the AS and the IR signals as u_0 and u_1, respectively. Only ρ_1 (dashed arrows in Fig. 4) is allowed to change through plasticity while ρ_0 (solid arrows connecting the AS neuron with learner neurons in Fig. 4) is set to a positive value, i.e., $\rho_0 = 1.0$. The output activity v of $\mathrm{L_n}$ and the learning rule for the weight change ρ_1^n are given by:

$$v(L_n) = \rho_0^n u_0 + \rho_1^n u_1, \quad n = 1, \ldots, 6, \tag{1}$$

$$\frac{d\rho_1^n}{dt} = \mu_n u_1 \frac{du_0}{dt}, \quad n = 1, \ldots, 6; \tag{2}$$

where we here use only input signals and correlate them with each other [31]. μ_n is the learning rate which is independently set for each learner neuron.

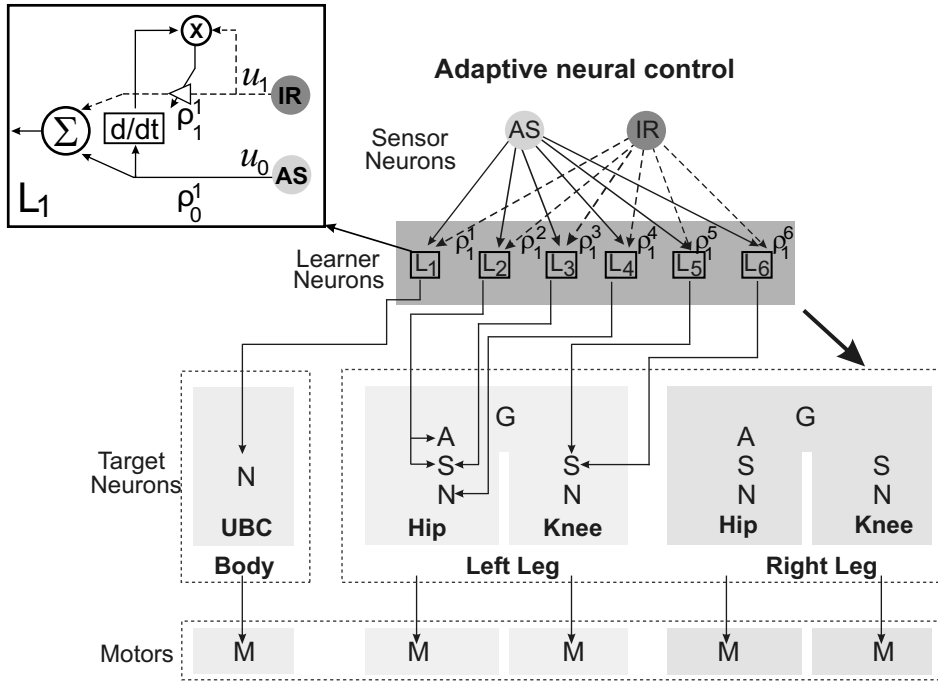

Fig. 4. Adaptive neural control where the neural learning mechanism is shown in the solid frame (top left, L_1, see text for details). Note that all learner neurons have the same learning mechanism. Connections between learner neurons and target neurons of the right leg, which are identical to those of the left leg, are not shown. ρ_1^1,...,ρ_1^6 are synaptic weights connecting the predictive IR signal with the learner neurons $(L_1, ..., L_6)$.

Here, the learning rate defines how much gait and body leaning of RunBot will change after each fall. It also implies to how fast the system can learn. For example, using small values requires several falls until legs and body responses are appropriately altered and vice versa for large values. One could also consider the learning rate as the susceptibility for a synaptic change, which in a biological agent will be defined by its evolutionary development, which determines the agent's ability to learn a certain task. How and if these values could also be influenced (possibly by mechanisms of meta-plasticity), changing learning susceptibility, goes beyond the scope of this article. Thus we empirically set it for the walking experiments presented below.

In neurons with multiple inputs this learning mechanism can be used to modify the synaptic strengths according to the order of the arriving inputs. As a consequence, the predictive input will get strengthened if the predictive signal u_1 is followed by the reflex input u_0, where the reflex drives the neuron into firing. This rule will lead to weight stabilization as soon as $u_0 = 0$ [31], hence, when the reflex has successfully been avoided. As a result, we obtain behavioral and synaptic stability at the same time without any additional weight-control mechanisms.

All in all, through the tight coupling of the mechanics with the adaptive reflex neural locomotion control, RunBot can autonomously walk with a high speed (> 3.0 leg length/s), self-adapting to minor disturbances, and reacting in a robust way to abruptly induced gait changes. At the same time, it can learn walking on different terrains, requiring only few learning experiences. All these experimental results have been presented in [22].

3 Experiment 1: Efference copy for external and self-generated sensory noise cancellation

As described above, RunBot uses IR (infrared eye) and AS (vestibular) information for posture and gait adaptation during walking up a painted slope. Due to the IR sensor characteristic, the sensor responds more strongly to the white color (see Fig. 2b). Thus, in our first experimental study on the application of the efference copy for locomotion control, the walking path of RunBot is modified by adding white spots on its black level tracks (compare Figs. 2b and 5a) in order to simulate disturbances to the IR sensor. As a consequence, the IR sensor gives unwanted periodic noise (see Fig. 5b, gray areas). In addition, RunBot's egomotion causes the AS to produce self-generated sensory events (see Fig. 5c, gray areas). These periodic perturbations will destabilize the activation parameters for the gait and lead to a wrong set of gait parameters as well as an inappropriate posture of the UBC. In other words, after a few learning experiences for walking up the slope, RunBot will perform upslope gait with leaning its UBC forwards during walking on level floors (location (1) or (3) shown in Fig. 5a). As a consequence, it will fall forwards before approaching a slope or after leaving it (see Sect. 3.2 for experimental results).

To solve such problems, we need to filter the unwanted noise. By doing so, we copy the periodic motor signals, transform them into noise expectation through so-called neural forward models (see Fig. 6). These expected sensory noise signals are fed into compensator units (see Fig. 6) to subtract the unwanted noise from the actual sensory feedback. Finally, we use neural post-processing units (see Fig. 6) to smooth and shape the compensated signals of the IR and AS sensors in order to obtain appropriate correlations for the learning mechanism. The details of this noise cancellation process is described in the following.

3.1 Modeling noise cancellation circuits

To filter the unwanted noise of the IR signal, we copy all extensor and flexor motor signals (N_R, N_L, efference copy) of the leg joints (see Fig. 6). These

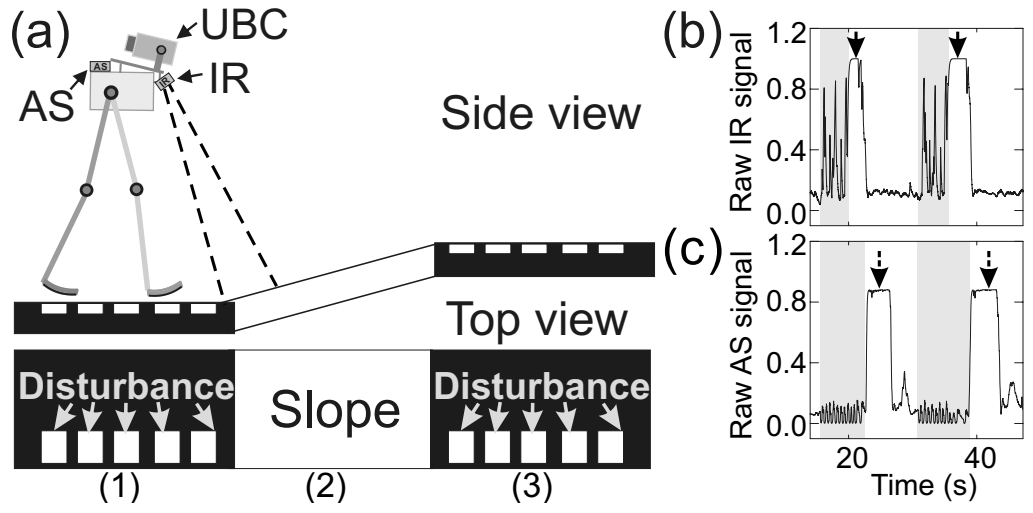

Fig. 5. (a) Walking path on which white spots are added at positions (1) and (3). They lead to a disturbance of the IR sensor. Note that the spots are empirically placed in the way that the IR sensor generates the unwanted noise every second step; i.e., periodic noise. We therefore set the distance between each spot to 15 cm (see Fig. 14a). Adding more spots (high density), the sensors will give continuous noise which makes the system impossible to discern between a slope detection signal and this continuous noise. (b, c) Raw sensor signals. Solid arrows in (b) depict the situation where RunBot detects a slope and dashed arrows in (c) where RunBot falls backwards. It falls over backwards, as it has not yet learned to react to its IR input with a change in gait.

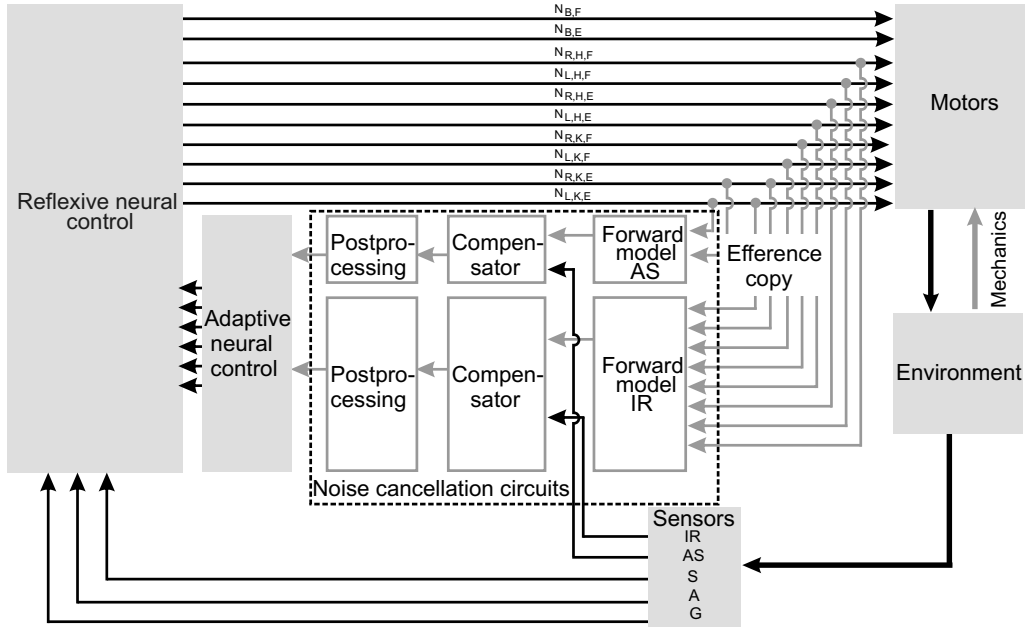

Fig. 6. Adaptive reflex neural locomotion control with external and self-generated sensory noise cancellation circuits. It applies an efference copy for eliminating sensory noise (compare Fig. 7).

motor signals are then transformed into a noise expectation through the neural forward model (forward model IR, see Figs. 6 and 7). This forward model is manually constructed as a series of 12 hysteresis elements [2] (see Fig. 7). Apart from filtering the noise it shapes the motor signals to match to the noise of the IR signal for subtraction afterwards. We create a hysteresis by using a single neural unit with a "supercritical" self-connection ($w_{Self} > 4$) [30], modeled as a discrete-time, rate-coded neuron with activity that develops according to:

$$a_i(t+1) = \sum_{j=1}^{n} W_{ij}\sigma(a_j(t)) + \Theta_i \quad i = 1, \ldots, n \tag{3}$$

where n denotes the number of units, a_i their activities, Θ_i represents a fixed internal bias term together with a stationary input to neuron i, and W_{ij} the synaptic strength of the connection from neuron j to neuron i. The output of the neurons is given by the standard sigmoid $\sigma(a_i) = (1 + e^{-a_i})^{-1}$. Input units are linearly mapped onto the interval [0, 1].

The neural parameters of the forward model network were empirically adjusted as follows. First we combined all motor signals at the first recurrent neuron $\mathrm{F1_{IR}}$ (see Fig. 7) and then we adjusted the combined motor signals such that they will cross forward and backward through the hysteresis domain [21], [30] for mainly filtering the noise of the motor signals. Hence, we set the synaptic weight, connecting between all motor signals and the recurrent neuron $\mathrm{F1_{IR}}$, to a positive value, i.e., 3.35, to amplify the signals. Afterwards, we shifted the amplified signals by a negative bias term, i.e., -6.3. Consequently, the modified signals sweep over the input interval between -6.3 and -2.95. Finally, we tuned the self-connection weight of the neuron to derive a reasonable hysteresis interval (see Fig. 8a) on the input space; i.e., 8.8. This hysteresis effect allows the output to show high (≈ 1.0) and low (≈ 0.0) activations at different points (see Fig. 8a). By utilizing this feature, the recurrent hysteresis neuron $\mathrm{F1_{IR}}$ acts as a low pass filter, which can eliminate unwanted motor noise (see Fig. 8).

After that, the output of the recurrent neuron $\mathrm{F1_{IR}}$ is provided to a series of single recurrent neurons $\mathrm{F2_{IR}}, ..., \mathrm{F12_{IR}}$ (see Fig. 7). The structure of each single recurrent neuron was configured in the same manner as the recurrent neuron $\mathrm{F1_{IR}}$ but the neural parameters were set differently. We chose them in the way that they provide the hysteresis effect (see Fig. 9a) that will shape

[2] Basically according to hysteresis effects, each recurrent neuron will prolong the activation time of its input signal, such that the more neurons are connected in a series, the longer the activation time of the output will be. Thus the neuron number of this forward model IR and other networks presented later is empirically set in a way that each recurrent neuron extends the response time of its input to finally match with the sensory noise for subtraction as well as to obtain a continuous and smooth signal.

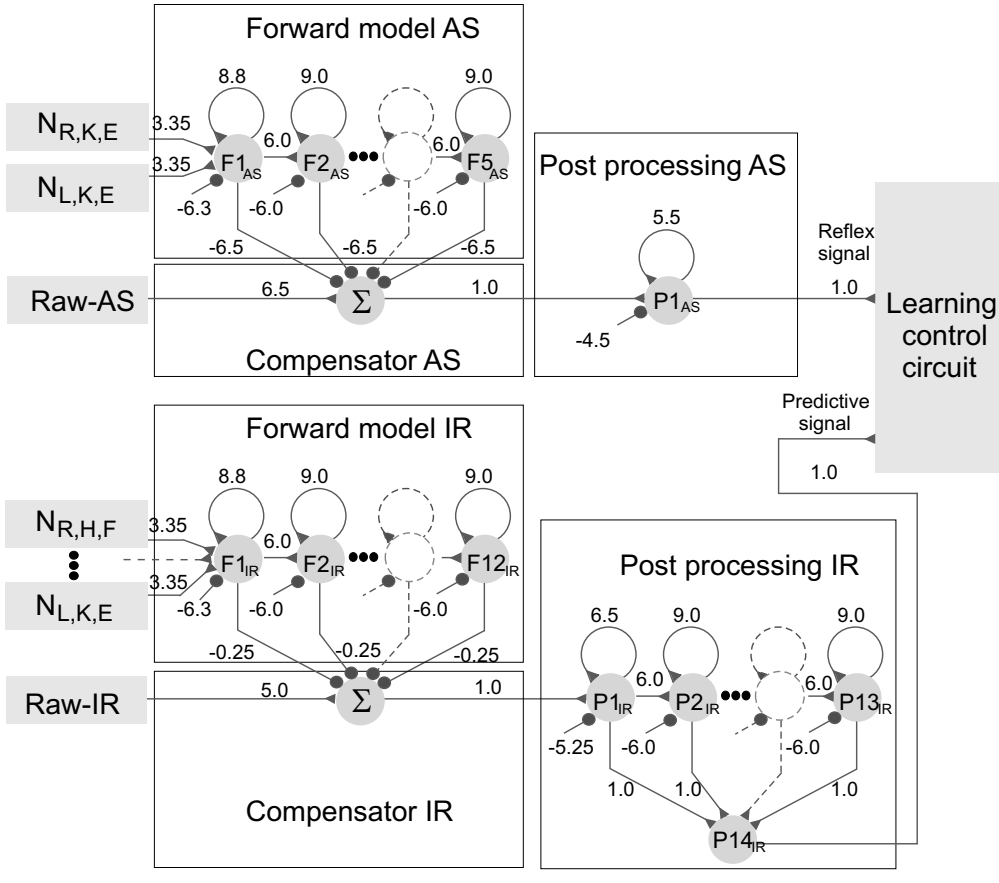

Fig. 7. Noise cancellation circuits of IR and AS signals. Each circuit is composed of three subunits: forward model, compensator, and postprocessing units. N_R and N_L indicate motor signals of leg joints (efference copy). Note that one can optimize these noise cancellation circuits, for instance by using an evolutionary algorithm [12], but for the purposes here, manual adjustment was sufficient.

the filtered motor signals to match the periodic noise of the IR signal. As a result, the connection weight between neurons, the bias term, and the self-connection weight are set as 6.0, -6.0, and 9.0, respectively. Eventually, the output of each recurrent neuron ($F1_{IR}, ..., F12_{IR}$) is transmitted to subtract the unwanted noise of the actual IR sensory feedback at a compensator unit (compensator-IR, see Fig. 7) through a connection weight set to -0.25. The compensator unit is simply modeled as a standard additive neuron with a linear transfer function. Subsequently, the compensator output is postprocessed at another series of recurrent neurons. All neural parameters of this postprocessing unit (postprocessing-IR, see Fig. 7) are set to similar values as $F2_{IR}, ..., F12_{IR}$, described above, except the first unit ($P1_{IR}$). Its neural parameters are given as: connection weight from the compensator unit IR to this first neurons = 1.0, bias term = -5.25, and the self-connection weight = 6.5. This postprocessing unit will smooth the signal at the recurrent neuron $P1_{IR}$ and through the remaining recurrent neurons $P2_{IR}, ..., P13_{IR}$ it will derive the

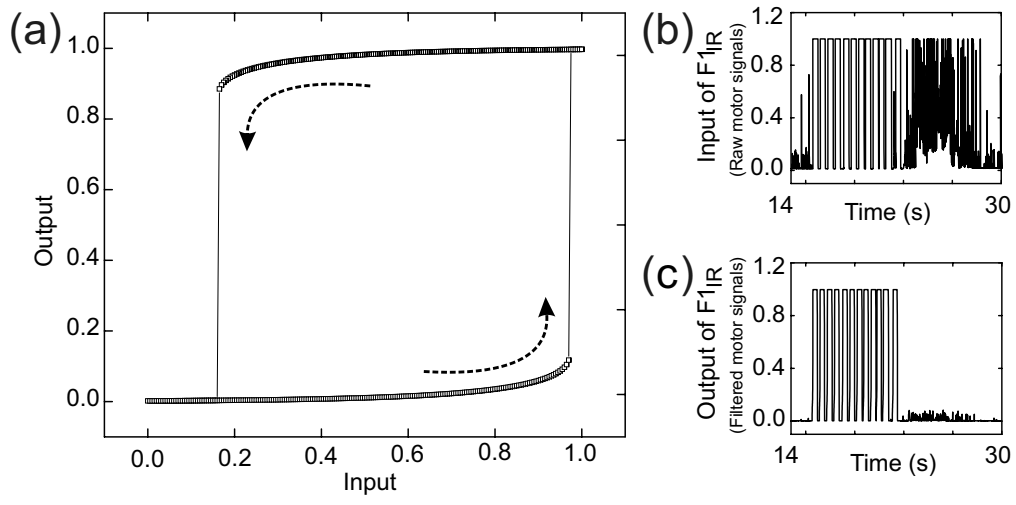

Fig. 8. (a) Hysteresis effect between the input and output of the recurrent neurons $F1_{IR}$ and $F1_{AS}$ (see Fig. 7). The input varies between 0.0 and 1.0 while the output shows high activation when the input increases to values above 0.97. On the other hand, it will show low activation when the input decreases below 0.165. Utilizing this hysteresis property, high frequency motor noise is eliminated. In other words, these recurrent neurons $F1_{IR}$ and $F1_{AS}$ act as a low pass filter. (b) Raw motor signals. (c) Filtered motor signals after passing through the recurrent neuron $F1_{IR}$. Note that the raw and filtered motor signals at the recurrent neuron $F1_{AS}$, having similar patterns to those of (b) and (c), are not shown.

appropriate correlation with the reflex signal for our learning mechanism. The final output of each postprocessing neuron is then summed up at the neuron $P14_{IR}$ before applying to the learning circuit as a predictive signal.

So far we have discussed the filtering process of the IR signal. To cancel self-generated noise at the AS signal, we use the same technique as above but here only the extensor knee-motor signals of the left ($N_{L,K,E}$) and right ($N_{R,K,E}$) legs are copied. Then they are transmitted to the neural forward model (forward model AS, see Figs. 6 and 7). Here the forward model AS consists of five recurrent neurons $F1_{AS}, ..., F5_{AS}$ (see Fig. 7). They are configured similar to the ones of the forward model IR. As a consequence, they lead to the same hysteresis phenomena (see Figs. 8a and 9a) which are used to filter motor noise and also transform the motor signals into the expected sensory noise signal in order to subtract the unwanted noise from the actual AS sensory feedback (self-generated sensation). The subtraction is done in the compensator unit (compensator-AS) where the output of each recurrent neuron is amplified and sent to this compensator-AS by means of a connection weight of -6.5. Note that the compensator-AS is modeled similar to the compensator-IR. Finally, the compensator output is shaped at the recurrent neuron $P1_{AS}$ (postprocessing-AS, see Fig. 7) before applying to the learning circuit as a reflex signal. The neural parameters of this postprocessing unit are set as: connection weight from the compensator-AS to its postprocessing

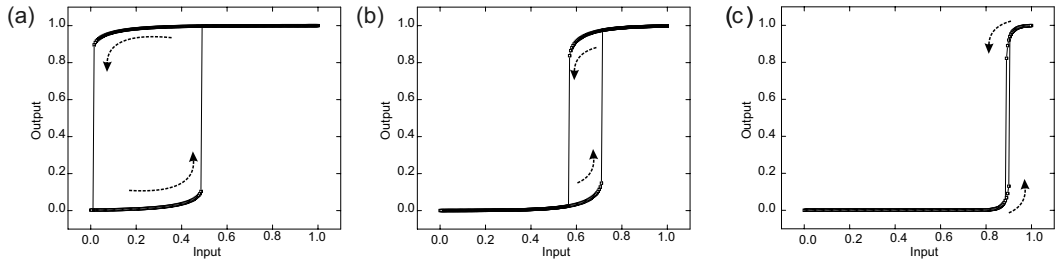

Fig. 9. Hysteresis diagrams of different recurrent neural parameters. All hystereses have an input which varies between 0.0 and 1.0 while their output shows low and high activations at different points. (a) Hysteresis loop of the recurrent neurons $F2_{IR}, ..., F12_{IR}$, $F2_{AS}, ..., F5_{AS}$, and $P2_{IR}, ..., P13_{IR}$ (see Fig. 7). The output shows high activation when the input increases to values above 0.49 while it will show low activation when the input decreases below 0.015. We use this hysteresis property for prolonging the activation time to obtain appropriate correlation of the signals, i.e., matching between the unwanted sensory noise and motor signals as well as correlating between the reflex (noise free AS) and predictive (noise free IR) signals. (b) Hysteresis loop of the recurrent neuron $P1_{IR}$ (see Fig. 7). The output shows high activation when the input increases to values above 0.71. On the other hand, it will show low activation when the input decreases below 0.57. This hysteresis effect is applied to smooth the output of the compensator-IR. (c) Small hysteresis loop of the recurrent neuron $P1_{AS}$ (see Fig. 7). Its output gives high activation when the input increases to values above 0.9 and it will show low activation when the input decreases below 0.88. This hysteresis effect is for smoothing the output of the compensator-AS.

neuron = 1.0, bias term = -4.5, and self-connection weight = 5.5. With these neural parameters, this postprocessing unit shows an appropriate hysteresis loop (see Fig. 9c) for refining the signal and providing proper correlation with the predictive signal for our learning mechanism.

3.2 Results

Here we firstly show the experimental results where the noise cancellation circuits were not applied to the control. As a consequence, the sensory noises destabilize the activation parameters for the gait after a few learning experiences because the synaptic connections (ρ_1^1,...,ρ_1^6, dashed arrows in Fig. 4) between the IR signal and the sites of movement control get strengthened. Once these connections are established, RunBot will react to its IR input with a gait change as soon as it gives a high activation value (either detecting the white slope or the white spots on the floors). In addition, the weights also show small glitches arising from the self-generated noise of the AS. Such glitches lead to a weak correlation with the IR signal and to minor weight changes (see Fig. 10). As a result, these perturbations make RunBot change its gait and UBC posture and make it fall forwards during walking (see Fig. 10). For

this demonstration, we refer the reader to the video clip at
http://www.bccn-goettingen.de/Members/CNgroup/runbot/efferencecopy.mpg.

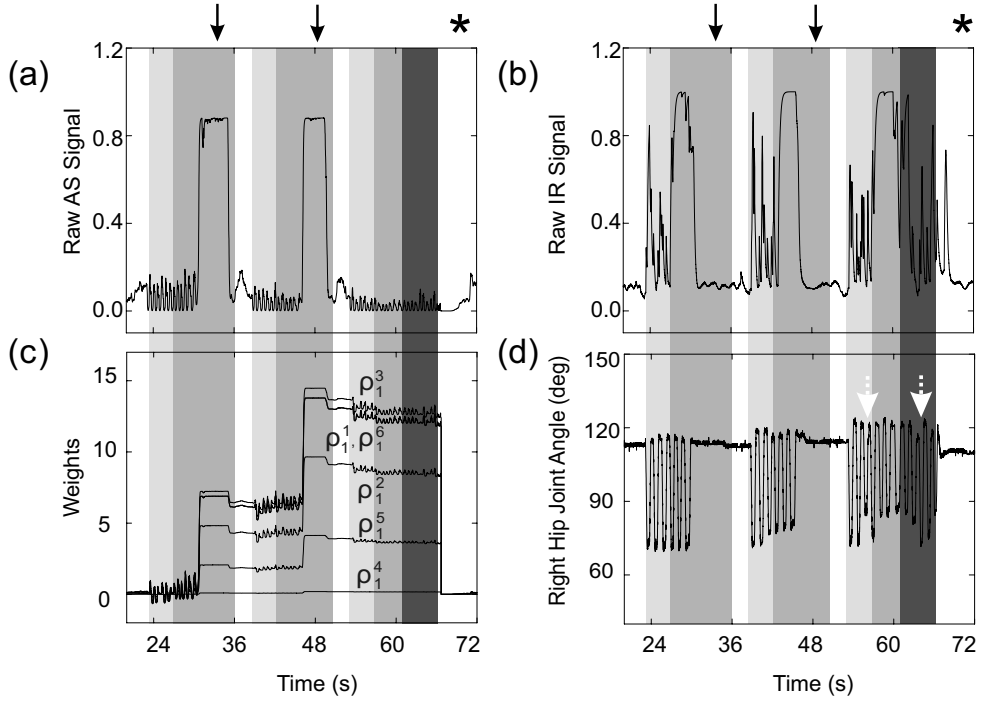

Fig. 10. Real-time data of an adaptive walking experiment when the noise cancellation circuits were not applied to the control. (a), (b) Raw sensor signals. (c) Growing synaptic strengths during the learning phase showing small glitches arising from the self-generated noise of the AS. (d) Right hip joint angle for all situations. The data was recorded while RunBot was initially walking from a lower floor (light gray areas) to an upper floor (dark gray areas) through a ramp (gray areas). Arrows depict the situation where RunBot falls backwards and white areas where RunBot was manually returned to the initial position. Dashed white arrows indicate the situation where RunBot's gait was disturbed. $*$ means the situation at which RunBot falls forwards. Note that here RunBot performs a few steps before falling forwards while walking on the upper floor. In this walking experiment we set the learning rate of each learner neuron (Eq. 2) as $\mu_1 = 10.0$, $\mu_2 = 7.0$, $\mu_3 = 10.5$, $\mu_4 = 0.14$, $\mu_5 = 3.0$, $\mu_6 = 10.0$.

On the other hand, when the noise cancellation circuits (see Fig. 6) described above are employed, the external and self-generated sensory noises [3] are eliminated (see Fig. 11). Thus RunBot can successfully learn to walk up an eight-degree painted slope after 3-5 falls and stably adapts its gait for walking on different terrains, i.e., level floors versus up the slope. For this demonstration, we refer the reader to the video clip at
http://www.bccn-goettingen.de/Members/CNgroup/runbot/efferencecopy.mpg.

[3] Recall that the external noise occurs from responding of the IR sensor to white spots placed on the level floors (see Fig. 5a) while the self-generated noise comes from the AS due to RunBot's ego motion.

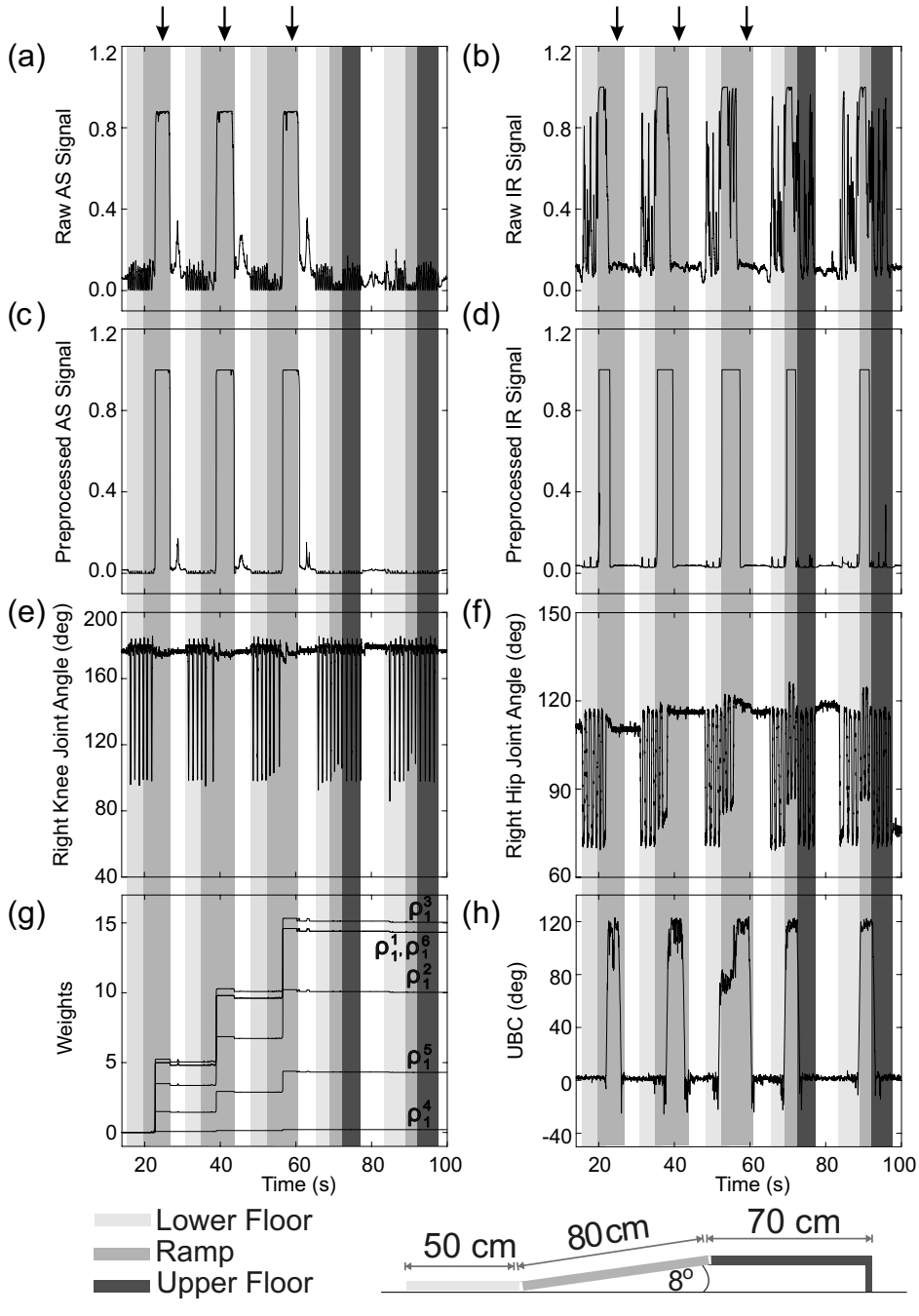

Fig. 11. Real-time data of an adaptive walking experiment when the noise cancellation circuits were applied to the control. (a), (b) Raw sensor signals. (c), (d) Corrected sensor signals showing a clear improvement. (e), (f) Right knee and hip joint angles for all situations. (g) Growing synaptic strengths during the learning phase. (h) Posture of the UBC where 0 degree means learning backwards while 120 degrees means leaning forwards. The data was recorded while RunBot was initially walking from a lower floor (light gray areas) to an upper floor (dark gray areas) through a ramp (gray areas). Arrows depict the situation where RunBot falls backwards and white areas where RunBot was manually returned to the initial position. Note that in this walking experiment we set the learning rate of each learner neuron (Eq. 2) as $\mu_1 = 10.0$, $\mu_2 = 7.0$, $\mu_3 = 10.5$, $\mu_4 = 0.14$, $\mu_5 = 3.0$, $\mu_6 = 10.0$.

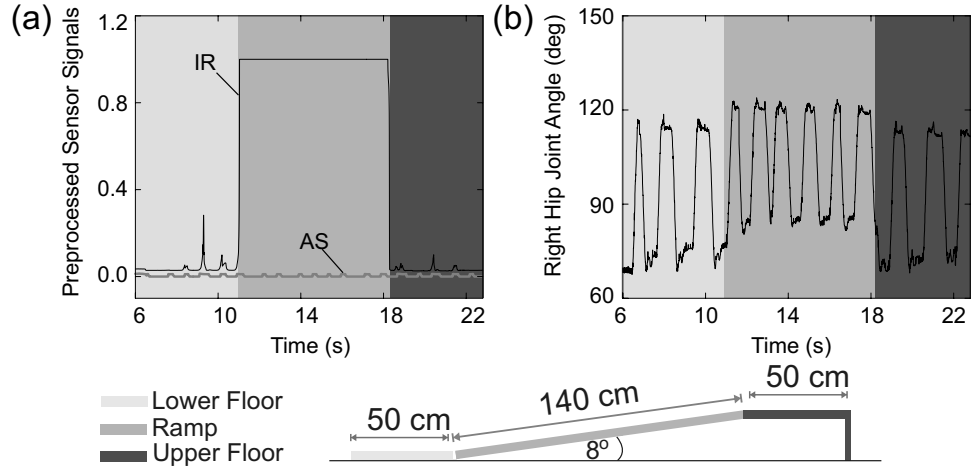

Fig. 12. Real-time data of an adaptive walking experiment on a 140 cm long slope. (a) Raw sensor signals. (b) Right hip joint angle during walking from a lower level (light gray area) to an upper floor (dark gray area) through a long ramp (gray area). Note that in this walking experiment we set the learned weights according to the ones shown in Fig. 11g.

After learning RunBot was also tested on the longer eight-degree ramp (140 cm long) to show the robustness of the system. As a result, it could manage to adaptively walk on this path without falling (see Fig. 12). This shows that the length of the slope does not have affect the stability of the system. In order to demonstrate more learning performance of the system, we also configured the ramp with different angles, e.g., four and twelve degrees. These different slope angles certainly require different amounts of learning. That is RunBot can manage to successfully walk up the four-degree ramp after 2 falls (see Fig. 13a) while the steeper one requires a larger UBC mass and 4 falls (see Fig. 13b).

Finally, to show the robustness of the noise cancellation circuits (Fig. 7) when different environmental conditions are given resulting in different kinds of IR sensory noise, we changed the distance X between the white spots on the black level tracks of RunBot (see Fig. 14). The performance of the circuits is shown in Fig. 14. It can be seen that the noise cancellation circuits are robust against such different kinds of IR sensory noise.

4 Experiment 2: Efference copy for slope detection

Due to gravitation, which excerts a different effect during walking up a slope, RunBot's forwards motion will be resisted. Consequently, the gait period of its walking cycle will be enlarged. This can be measured at the motor signals (see Fig. 15) because they are basically derived from the proprioceptive feedback, i.e., foot contact and joint angle sensors.

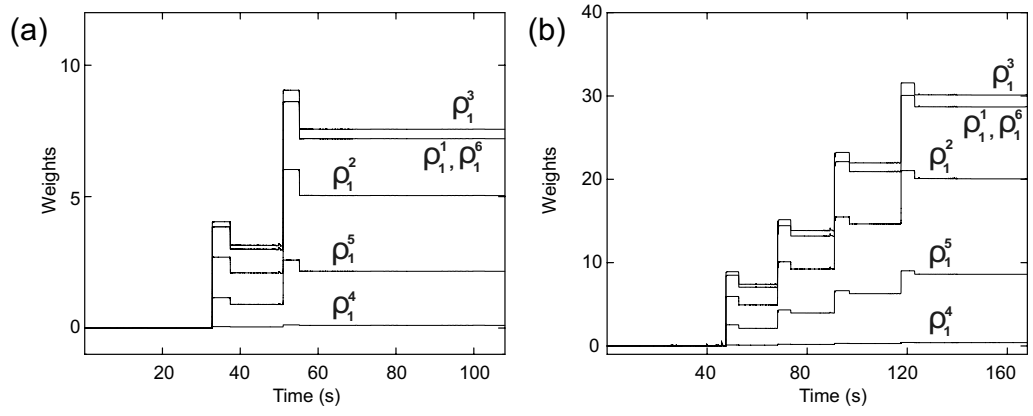

Fig. 13. (a) Growing synaptic strengths during learning to walk up a four-degree painted slope with 80 cm length (compare to Fig. 11g). Note that two steps of synaptic changes imply two falls of RunBot. (b) Growing synaptic strengths during learning to walk up a twelve-degree slope (compare to Fig. 11g). Note that four steps of synaptic changes imply four falls of RunBot. In both cases, the sensory signals and the changes of leg and body movements develop in a similar way as that shown in Fig.11 while only the amplitudes of the movements during walking on the different ramps are different due to the learned weights that convert to different values. Such that, in comparison to the leaning motion of the legs and body for the eight-degree ramp, the legs and body lean less forwards for the four-degree ramp while they lean more forwards for the twelve-degree ramp.

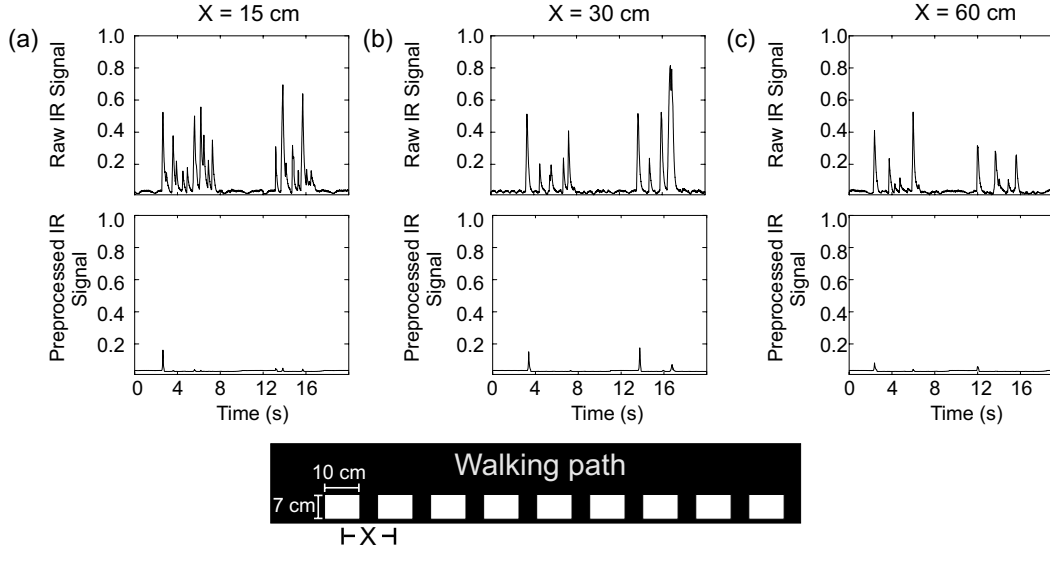

Fig. 14. (a) Raw and preprocessed IR signal where the distance X between the white spots is 15 cm. (b) Raw and preprocessed IR signal where the distance X between the white spots is 30 cm. (c) Raw and preprocessed IR signal where the distance X between the white spots is 60 cm. Note that when placing the white spots closer than 15 cm to each other (high density), the sensors will give continuous noise which makes the system impossible to discern between a slope detection signal and this continuous noise. The experimental setup is shown in the lower picture.

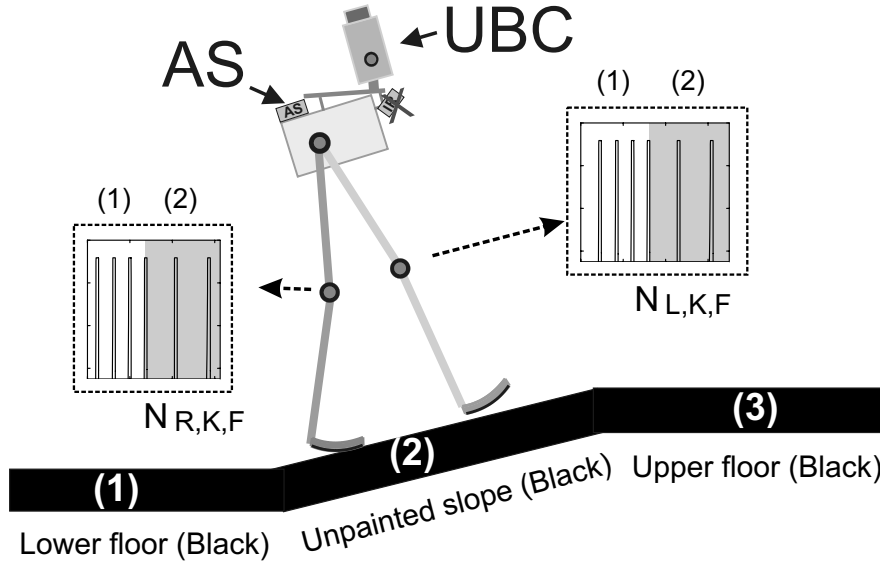

Fig. 15. Observed flexor-knee motor signals of the left ($N_{L,K,F}$) and right ($N_{R,K,F}$) legs during walking from a lower floor (1) to an eight-degree unpainted slope (2). They show that the gait period of RunBot's walking cycle is increased (gray areas in windows) while walking on the slope. In this situation, RunBot walks a few steps on the slope before it falls over backwards, as it has not yet learned to detect the slope and react to it with a change in gait and its UBC posture. Note that for this experiment we set its UBC posture in a more upright position (75 degrees) in order to allow it to walk a few steps on the slope before falling backwards.

According to this effect, the experimental study here will show the use of only the motor signals for discerning an unpainted slope (see Fig. 15). This can replace the use of the IR signal, where the slope needs to be painted as shown in the previous experiment. However, before applying the motor signals as a predictive signal to our learning mechanism (see Fig. 4), they need to be transformed into a signal (called slope detection signal) which can appropriately correlate with the reflex signal (AS). Hence a so-called slope detection circuit is developed and employed for this purpose (see Fig. 16) as described in the following.

4.1 Modeling a slope detection circuit

To obtain the slope detection signal derived from the efference copy, we use here the knee flexor-motor signals of the right $N_{R,K,F}$ and left $N_{L,K,F}$ legs and feed them into the motor signal transformation circuit (see Fig. 17), which was empirically constructed. It consists of 15 neurons $M1, ..., M15$ where the neural parameters of $M1, ..., M14$ together with their function are similar to those of the neural forward model IR. That is the first recurrent neuron filters the motor noise while the rest shape the motor signals by prolonging their

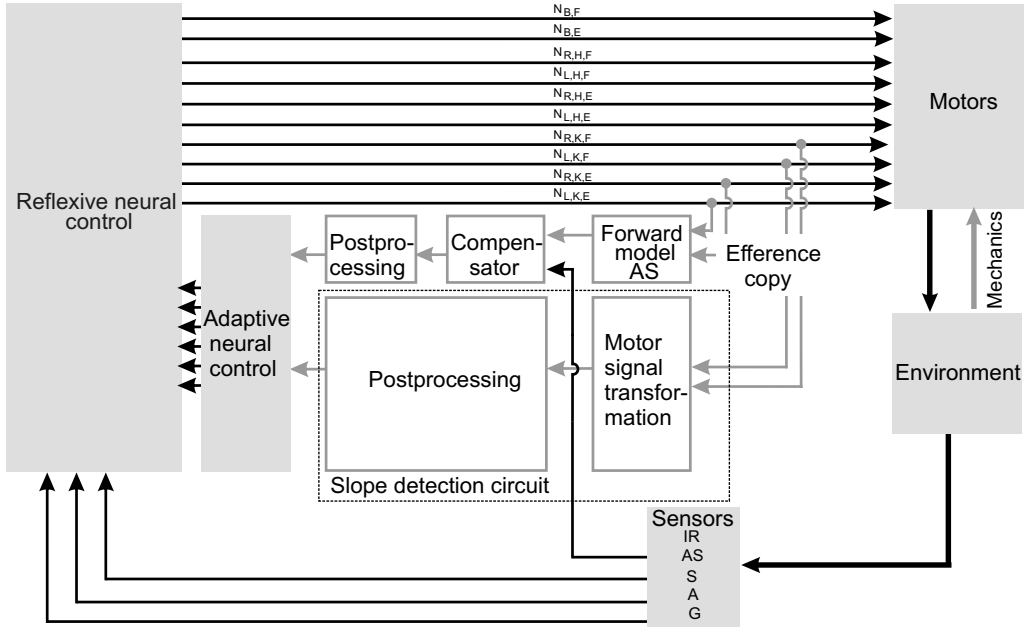

Fig. 16. Adaptive reflex neural locomotion control with noise cancellation and slope detection circuits. This control enables RunBot to detect an unpainted slope through the change of its own motor signals rather than through the IR signal. As a result, it can successfully learn to walk up the slope by utilizing only AS and motor information.

activation time. Eventually the output of each recurrent neuron is combined at the neuron M15. Consequently, the pulse shaped motor signals will become continuous and smooth. In other words, they show continuous high activation (≈ 1) during walking on level floors rather than spikes. But they will show a drop to about zero of their continuous activation while walking on the slope due to increasing the gait period of its walking cycle or they will become completely deactivated if RunBot falls backwards on the slope. Such a drop and/or deactivation enables the system to recognize the slope. However we need to convert this into a positive value for correlation with the AS signal in our learning mechanism. Thus, a postprocessing unit is used to invert the signal via neuron $P1_M$. As a result, the drop and deactivation will turn into a positive value (≈ 1) while the continuous high activation will become zero. Subsequently, the recurrent neuron $P2_M$ enlarges the response time of the inverted drop and deactivation signals. The resulting signal will be sent to the learning control circuit, which allows RunBot to learn to walk up the unpainted slope. Note that all neurons are modeled as additive neurons with a standard sigmoid transfer function (Eq. 3) except the neuron $P2_M$ which has a linear transfer function with its output restricted to the interval [0, 1].

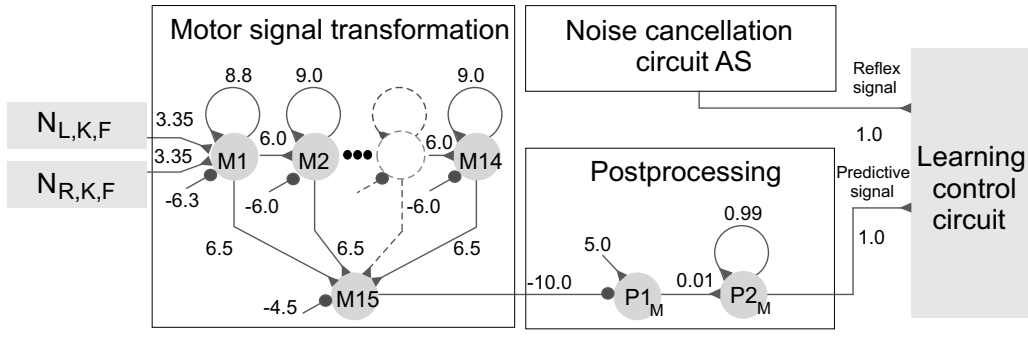

Fig. 17. Slope detection circuit. It consists of two subunits: motor transformation and postprocessing units. In the postprocessing unit the neuron $P1_M$ performs as a signal inverter while the recurrent neuron $P2_M$ with a linear transfer function acts as a low pass filter. $N_{L,K,F}$ and $N_{R,K,F}$ indicate motor signals of the left and right knee flexors, respectively. Note that one can optimize this slope detection circuit, for instance by using an evolutionary algorithm [12], but for the purposes here, manual adjustment was sufficient.

4.2 Results

Figure 18 shows experimental results where the motor signals together with the slope detection circuit described above is employed (see Fig. 16) instead of using the IR signal for discerning a slope. As a consequence, RunBot can successfully learn to walk up an eight-degree unpainted slope after 2-5 falls. After that it can stably adapt its gait and UBC for walking on different terrains, i.e., level floors versus up the slope. For this demonstration, we refer the reader to the video clip at
http://www.bccn-goettingen.de/Members/CNgroup/runbot/efferencecopy.mpg.
Walking on different angles of the slope will certainly require different amounts of learning while the length of the slope will not disturb the stability of the system (not shown but compare to Figs. 12 and 13).

5 Discussion and conclusions

Here, we concisely discuss and conclude some remaining issues following the presented experiments while most of the relevant discussion points have been described alongside the experimental sections above. In this study, we have addressed the exploitation of an efference copy together with internal models for dynamic locomotion control in terms of sensory processing and terrain determination. The first experiment has shown the relationship between afference (sensory information) and efference (motor command). That is a copy of the efference after modification through neural forward models is used to subtract external and self-generated sensory noise (sensory processing) in order

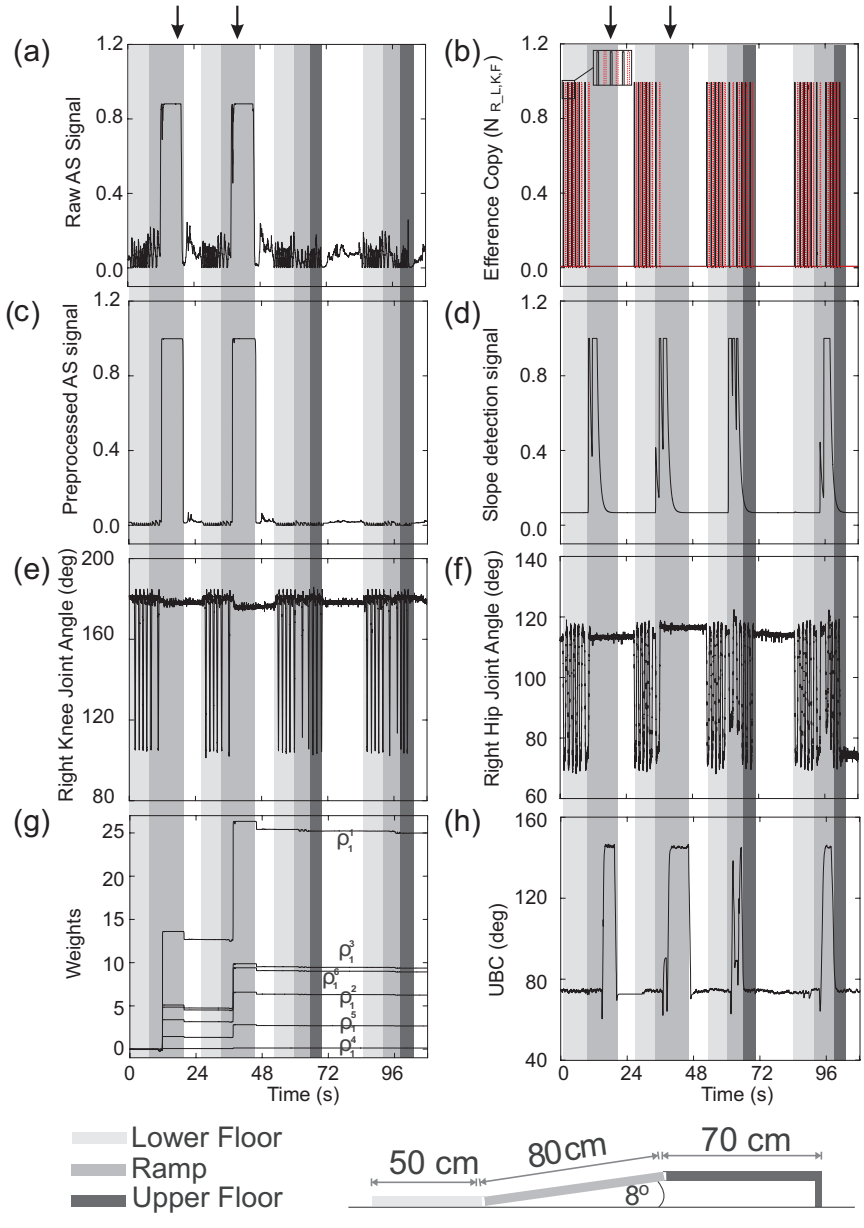

Fig. 18. Real-time data of an adaptive walking experiment where RunBot detects an unpainted slope through its motor signals. (a) Raw AS signal. (b) Knee flexor-motor signals of the right $N_{R,K,F}$ (solid line) and left $N_{L,K,F}$ (dashed red line) legs used for slope detection (see also in a zoom window). (c) Preprocessed AS signal (reflex signal). (d) Slope detection signal (predictive signal). (e), (f) Right knee and hip joint angles for all situations. (g) Growing synaptic strengths during the learning phase. (h) Posture of the UBC. It leans 75 degrees as an initial upright position, while 145 degrees means leaning forwards. In this experiment, RunBot can successfully walk up the slope after two learning experiences. The data was recorded while RunBot was initially walking from a lower floor (light gray areas) to an upper floor (dark gray areas) through a ramp (gray areas). Arrows depict the situation where RunBot falls backwards and white areas where RunBot was manually returned to the initial position. Note that in this walking experiment we set the learning rate of each learner neuron (Eq. 2) as $\mu_1 = 14.0$, $\mu_2 = 3.5$, $\mu_3 = 5.25$, $\mu_4 = 0.07$, $\mu_5 = 1.5$, $\mu_6 = 5.0$.

to obtain perceptual stability for correctly guiding locomotion. Furthermore, we have also shown that the employed forward model IR is robust against different IR noise characteristics.

In the second experiment, we have demonstrated that efference copy signals derived from a reflexive mechanism [22] are capable of determining terrain condition changes, e.g., level floor versus up a slope. Due to gravitation, which exerts a different effect during walking up a slope, RunBot's forwards motion will be resisted. In other words, the gait period of its walking cycle will be enlarged which can be measured at the motor signals.

In both walking experiments, leaning the UBC forward and changing several gait parameters through a learning mechanism, RunBot manages to walk on different slope angles, e.g., 4, 8 and 12 degrees. While a steep slope (12 degrees) requires a larger UBC mass. On the other hand, walking down slopes can be also achieved in the reverse way with an appropriate gait (not shown here but see [23]). This is achieved by learning which is based on simulated plasticity. Moreover, walking on different slope lengths can be tackled where the length of the slope has no effect to the stability of the system.

The employed dynamic locomotion controller of RunBot in these experimental studies was modeled as artificial neural networks using discrete-time dynamics. The networks consist of three main components or modules: Reflexive network for basic walking, adaptive network for learning capability, and internal model building network [4] for transforming efference copy signals into desired sensory signals. All in all this leads to a certain higher complexity of the control structure. The three modules, however, can be understood one by one, which makes network design analyzable. In addition to this, we have shown that hysteresis effects of a single recurrent neuron could be utilized for designing forward model, motor signal transformation, and postprocessing units (internal model building units). Although most units consist of several single recurrent neurons in series (see Figs. 7 and 17), they give understandable and analyzable network characteristics which can be applied to other applications [11], [25], [24], [26] requiring high and low output activations at different points on the input space (hysteresis effects, see Figs. 8 and 9). If desired, all manually adjusted parameters of these units can be optimized by an evolutionary algorithm [12] or entire units can be constructed as feedforward neural networks by an error back-propagation technique [34]. However, it is important to note that, in principle, the single recurrent neuron used here, e.g., the neuron $F1_{IR}$, performs as a nonlinear low-pass filter which eliminates high frequency noise; i.e., here motor noise (see Fig. 8b), while the motor signal, which contains much lower frequencies can pass through. This way this neuron

[4] Here we call the noise cancellation circuits (see Fig. 7) and the slope detection circuit (see Fig. 17) as the internal model building network.

can be designed to have any cutoff frequency below the motor noise frequency but above the motor signal frequency. The cutoff frequency of this nonlinear filter neuron is mainly derived from its self-connection weight [26] and we find that the neural parameter, which control this (self-connection weight), can be adjusted within large intervals making any fine tuning unnecessary to obtain appropriate nonlinear low-pass filters in certain frequency ranges (see [26] for the parameter analysis).

To a certain extent the experimental studies pursued here sharpen our understanding of how the efference copy influences the dynamic locomotion control. They also emphasize how biological findings (efference copy and internal models) can be beneficially used in robotic systems. To date, efference copy and internal model concepts have been applied to a number of robot control problems in different approaches. For example, Namiki et al. (2003) [29] presented a hierarchical parallel control architecture for high-speed visual servoing (arm motion control system with visual perception). The architecture is based on an interaction model between efferent and afferent signals in a motor control network used for a parameter adaptation mechanism. As a consequence, it allows the robot to perform high-speed tracking, grasping, handling, and collision avoidance tasks. Russo et al. (2005) [33] simulated phonotaxis (auditory orientation towards sound sources) and an optomotor reflex (a visual capability compensating for external disturbances to maintain a straight trajectory) on a robot. The smooth integration of auditory and visual stimuli is achieved via a forward model. It takes acoustically driven motor command signals (efference copy) and tries to predict the reafferent visual signal such that the optomotor reflex is inhibited during phonotaxis behavior. In the domain of walking robot control, Lewis and Bekey (2002) [20] used innate internal models to transform an efference copy from a central pattern generator (CPG) unit into the sensory expectation. This expected sensory information is compared with the actual sensory feedback and an adaptive rule then modifies the CPG to coordinate the limbs of a quadruped robot. Dürr et al. (2003) [5] purposed a neural three-joint leg control mechanism for a hexapod robot for leg searching movement. In addition, they also provided a generalized form of their mechanism where the internal model and the efference copy are applied for central pattern control. Compared to many of these approaches, here we focus on showing the usefulness of the efference copy and internal model in dynamic locomotion control which, to the best of our knowledge, has not been investigated so far.

Although our approach cannot be directly related to how biological systems solve similar tasks, there is ample evidence suggesting that biological systems use efference copy and internal model mechanisms to maintain stable perception as well as to perform fast, robust, and adaptive behavior [3], [4], [36], [38]. For example, as described in [10], flying insects can discern self-generated sensation (i.e., rotation of the visual field caused by tracking a target) from ex-

ternal sensation due to changes in the external world (i.e., visual rotation due to air disturbances). This could be done by using motor outputs transformed into the expected visual inputs to suppress the self-generated sensation. In male grasshoppers, an auditory interneuron activity (G-neuron) is inhibited during stridulation (making a shrill sound by rubbing hind legs and wings) [37]. This is because proprioceptive feedback and efference copy signals of the hind legs act together to switch-off the interneuron response during stridulation. In crickets, interneurons sensitive to movement of the antennae give less activation during active motion by the cricket itself [7]. Another classic example is that moving our eyes causes the image on the retina to move, but we obtain stable image perception because the image movement is predictable from the eye movement command [27] (but see [2]). Furthermore, Cullen and Roy (2004) [32] showed that in primates vestibular signals arising from self-generated head motions are inhibited by an internal model mechanism for perceptual stability and accurate behavior control.

Finally, we would like to discuss our locomotion control network (see Figs. 6 and 16) with respect to the general concept of internal models [18], [39]. Internal models or internal loops, systems that imitate the behavior of a biological process, have appeared as an important theoretical concept in motor control. They are generally divided into two main categories [18], [39]: Inverse internal model and forward internal model. In addition, the combination of inverse and forward internal models is called integration of multiple internal models. The inverse internal model is a system that transforms a desired trajectory/state information into a motor command for generating movements. Such a model can be described as a controller. By contrast, the forward internal model is a system that predicts the next state (state estimation) and/or sensory consequences (expected sensory feedback) from the current state and motor command (efference copy). In other words, it can be viewed as a predictor. The integration of multiple internal models proposes that multiple pairs of inverse and forward internal models are tightly coupled as functional units. Additionally, from neurophysiological and biological studies, it is known that in movement control forward and inverse models involve the dynamics of the motor system changing under different conditions. Thus it has been suggested that the internal models must be adaptable and learnable by, e.g., supervised learning [14], feedback-error-learning [17], direct inverse modeling [19], and auto-imitatively adapting inverse modeling [15], [16].

Compared to the different types of the internal models described above, the reflexive neural network of our robot can be implicitly understood as the inverse model that calculates motor commands from sensory inputs rather than desired trajectories. Note that the RunBot system does not use any trajectory control, instead only a pure sensor-driven mechanism is employed [22]. On the other hand, the internal model building network is comparable to the forward internal model that estimates sensory feedback (see experiment

1) and walking state (see experiment 2) from motor commands. This internal network was designed as a non-adaptable network where a motor learning mechanism is executed separately in the adaptive neural network [5], which results in locomotor adaptation; i.e., adaptive walking on different terrains. More demanding tasks will be the improvement on the mechanical design of RunBot by integrating elastic and passive joint properties with novel types of motors including a spring mechanism to imitate muscle properties for better self-stabilization through mechanical feedback, called "preflexes" [6].

Acknowledgments

This research was supported by the PACO-PLUS project as well as by BMBF (Federal Ministry of Education and Research), BCCN (Bernstein Center for Computational Neuroscience)–Göttingen W3.

References

[1] T. Bem, J. M. Cabelguen, Ö. Ekeberg, S. Grillner, From swimming to walking: a single basic network for two different behaviors, Biological Cybernetics 88 (2003) 79–90.

[2] B. Bridgeman, Efference copy and its limitations, Computers in Biology and Medicine 37 (7) (2007) 924–929.

[3] J. M. Camhi, A. Levy, Organization of a complex movement: fixed and variable components of the cockroach escape behaviour, Journal of Comparative Physiology A 163 (1988) 317–328.

[4] K. E. Cullen, Sensory signals during active versus passive movement, Curr. Opin. Neurobiol. 14 (6) (2004) 698–706.

[5] V. Dürr, A. F. Krause, J. Schmitz, H. Cruse, Neuroethological concepts and their transfer to walking machines, International Journal of Robotics Research 22 (2003) 151–167.

[6] R. J. Full, D. E. Koditschek, Templates and anchors: Neuromechanical hypotheses of legged locomotion on land, Journal of Experimental Biology 202 (1999) 3325–3332.

[7] M. Gebhart, H. W. Honnegger, Physiological characterisation of antennal mechanosensory descending interneurons in an insect (Gryllus bimaculatus,

[5] Recall that the learning algorithm applies a correlation based differential Hebbian learning rule.

Gryllus campestris) brain, Journal of Experimental Biology 204 (2001) 2265–2275.

[8] T. Geng, B. Porr, F. Wörgötter, Fast biped walking with a sensor-driven neuronal controller and real-time online learning, The International Journal of Robotics Research 25 (3) (2006) 243–259.

[9] R. Held, Exposure history as a factor in maintaining stabilitity of perception and coordination, Journal of Nervous and Mental Disease 132 (1961) 26–32.

[10] E. v. Holst, H. Mittelstaedt, Das Reafferenzprinzip, Naturwissenschaften 37 (1950) 464–476.

[11] M. Hülse, S. Wischmann, P. Manoonpong, A. Twickel, F. Pasemann, Dynamical systems in the sensorimotor loop: On the interrelation between internal and external mechanims of evolved robot behavior, in: M. Lungarella, R. Pfeifer (eds.), Proceedings of 50 Years of Artificial Intelligence, vol. 4850, Springer-Verlag, 2007.

[12] M. Hülse, S. Wischmann, F. Pasemann, Structure and function of evolved neuro-controllers for autonomous robots, Connection Science 16 (4) (2004) 249–266.

[13] A. J. Ijspeert, A. Crespi, D. Ryczko, J. M. Cabelguen, From swimming to walking with a salamander robot driven by a spinal cord model, Science 315 (5817) (2007) 1416–1420.

[14] L. Jordan, Supervised learning and systems with excess degrees of freedom, Technical Report COINS 88/27 (1998) 1–41.

[15] K. T. Kalveram, The inverse problem in cognitive, perceptual and proprioceptive control of sensorimotor behaviour: Towards a biologically plausible model of the control of aiming movements, International Journal of Sport and Exercise Psychology 2 (2004) 255–273.

[16] K. T. Kalveram, A. Seyfarth, Learning the inverse model of the dynamics of a robot leg by auto-imitation, Springer Berlin Heidelberg, 2007.

[17] M. Kawato, Feedback-error-learning neural network for supervised motor learning, Advanced neural computers (1990) 365–372.

[18] M. Kawato, Internal models for motor control and trajectory planning, Curr. Opin. Neurobiol. 9 (1999) 718–727.

[19] M. Kuperstein, Neural model of adaptive hand-eye coordination for single postures, Advanced neural computers 239 (1998) 1308–1311.

[20] M. A. Lewis, G. A. Bekey, Gait adaptation in a quadruped robot, Autonomous Robots 12 (3) (2002) 301–312.

[21] P. Manoonpong, Neural preprocessing and control of reactive walking machines: Towards versatile artificial perception-action systems, Cognitive Technologies, Springer, 2007.

[22] P. Manoonpong, T. Geng, T. Kulvicius, B. Porr, F. Wörgötter, Adaptive, fast walking in a biped robot under neuronal control and learning, PLoS Computational Biology 3 (7) (2007) e134.

[23] P. Manoonpong, T. Geng, F. Wörgötter, Exploring the dynamic walking range of the biped robot "Runbot" with an active upper-body component, in: Proceedings of the Sixth IEEE-RAS International Conference on Humanoid Robots (Humanoids 2006), 2006.

[24] P. Manoonpong, F. Pasemann, F. Wörgötter, Neural preprocessing of auditory-wind sensory signals and modular neural control for auditory- and wind-evoked escape responses of walking machines, in: Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics, 2008.

[25] P. Manoonpong, F. Pasemann, F. Wörgötter, Sensor-driven neural control for omnidirectional locomotion and versatile reactive behaviors of walking machines, Robotics and Autonomous Systems 56 (3) (2008) 265–288.

[26] P. Manoonpong, F. Wörgötter, Simple recurrent neural filters for non-speech sound recognition of reactive walking machines, the Bernstein Conference on Computational Neuroscience (BCCN) (2009) Submitted.

[27] L. Matin, Eye movements and perceived visual direction, Springer, New York, 1972.

[28] J.-A. Meyer, S. Doncieux, D. Filliat, A. Guillot, Evolutionary approaches to neural control of rolling, walking, swimming and flying animats or robots, Physica-Verlag GmbH, Heidelberg, Germany, 2003.

[29] A. Namiki, K. Hashimoto, M. Ishikawa, A hierarchical control architecture for high-speed visual servoing, International Journal of Robotics Research 22 (10–11) (2003) 873–888.

[30] F. Pasemann, Dynamics of a single model neuron, International Journal of Bifurcation and Chaos 2 (1993) 271–278.

[31] B. Porr, F. Wörgötter, Strongly improved stability and faster convergence of temporal sequence learning by using input correlations only, Neural Computation 18 (6) (2006) 1380–1412.

[32] J. E. Roy, K. E. Cullen, Dissociating self-generated from passively applied head motion: neural mechanisms in the vestibular nuclei, J. Neuroscience 24 (9) (2004) 2102–2111.

[33] P. Russo, B. Webb, R. Reeve, P. Arena, L. Patane, A cricket-inspired neural network for feedforward compensation and multisensory integration, in: Proceedings of the 44th IEEE Conference on Decision and Control, and the European Control Conference 2005, 2005.

[34] J. Schröder-Schetelig, P. Manoonpong, F. Wörgötter, Using efference copy and neural control for adaptive walking on different terrains, Autonomous robots (2009) Submitted.

[35] R. Sperry, Neural basis of the spontaneous optokinetic response produced by vision inversion, Journal of Comparative and Physiological Psychology 43 (1950) 482–489.

[36] B. Webb, Neural mechanisms for prediction: Do insects have forward models?, Trends in Neurosciences 27 (2004) 278–282.

[37] H. Wolf, O. von Helversen, 'Switching off' of an auditory interneuron during stridulation in the acridid grasshopper Chorthippus biguttulus L., Journal of Comparative Physiology A 158 (1986) 861–871.

[38] D. M. Wolpert, Z. Ghahramani, Computational principles of movement neuroscience, Nature Neuroscience 3 (2000) 1212–1217.

[39] D. M. Wolpert, Z. Ghahramani, M. I. Jordan, An internal model for sensorimotor integration, Science 269 (5232) (1995) 1880–1882.

[40] D. M. Wolpert, M. Kawato, Multiple paired forward and inverse models for motor control, Neural Networks 11 (1998) 1317–1329.