Supplement for: From Adaptive Locomotion to Predictive Action Selection – Cognitive Control for a Six-Legged Walker

Authors

M. Schilling,[1]* J. Paskarbeit,[2] H. Ritter,[1] A. Schneider,[2] H. Cruse[3]

* - corresponding author: Malte Schilling, email: mschilli@techfak.uni-bielefeld.de

Technical Description

1) Simulator and Controller Implementation

The control framework and the simulator (next section) are publicly available (dynamical simulation environment is realized in C++ and based on the Open Dynamics Engine library, see https://github.com/malteschilling/hector ; the controller has been implemented in python (version 3), see https://github.com/malteschilling/cognitiveWalker). For more details on the body model see (Schilling et al., 2012); for the cognitive expansion see above and (Schilling and Cruse, 2017), which showed a proof of concept in simulation, and for the neural network for action selection see (Schilling, 2017).

2) Walknet

The decentralized control system Walknet has been described in detail in (Schilling et al., 2013a and 2013b). It is controlled by a neural network control structure that (as described in the method section) is distributed into individual controllers per each leg. Here, we briefly present a more formal description of such a single leg controller (for details see open repository with code implementation –MotivationNetLeg.py– and original publication in Schilling et al., 2013a,b).

Each leg controller consists of motivation units that modulate the different movement primitives (swing and stance, each for forward and backward walking). There is basically a competition between these units in a winter-take-all fashion and switching between behaviors is driven mainly by sensory inputs. A motivation unit is simply acting as a weighted summation unit with a non-negative and semi-linear activation function (saturating for values greater 1.). Therefore, activity of a unit is calculated at time *t* following:

$$a_j(t) = f(\sum_i w_i j * a_i(t-1) + bias_j), \text{with } f(x) = \max(0., \min(x, 1.))$$

The connections between units in a single leg are sparse and given in table S3. Upper row and left column refer to the individual motivation units. Weights arranged in a horizontal row mark input weights of the unit shown at the left (rec_w = 0.6). A simple schematic is shown in Fig. 3 (and S6) of the original article (there only a forward unit is shown and we didn't show the leg and backward unit). Units in italics represent sensory units or externally set (GC = ground contact, beh_PEP = measures if leg moved behind the assumed posterior extreme position as explained in the method summary for Walknet, all PDs = an integration of all problem detectors that detect instabilities and switch of the normal walking behavior). Last, sw_start is a unit with a temporal behavior that is active for 0.2 seconds in order to guarantee that swing was initiated before reactivating stance.

Table S3: Connection weights for a single leg controller.

	fw	bw	leg_MU	sw	sw_f	sw_b	st	st_f	st_b	*beh_PEP*	*GC*	sw_start	*All PDs*	bias
fw	*Driven by global input*													1
bw														
leg_MU			rec_w											
sw			0.25	rec_w	0.8	0.8	-0.75			6.75		6.75		
sw_f	-0.15			0.45	rec_w	-0.6							*Stop Walking and activate Cogn. Exp. = switch to search for behavior*	
sw_b		-0.15		0.45	-0.6	rec_w								
st			0.25	-0.75			0.25	0.8	0.8	-6.75	2.25	-6.75		
st_f	-0.15						0.45	rec_w	-0.6					
st_b		-0.15					0.45	-0.6	rec_w					
sw_start				1										

3) Cognitive Expansion

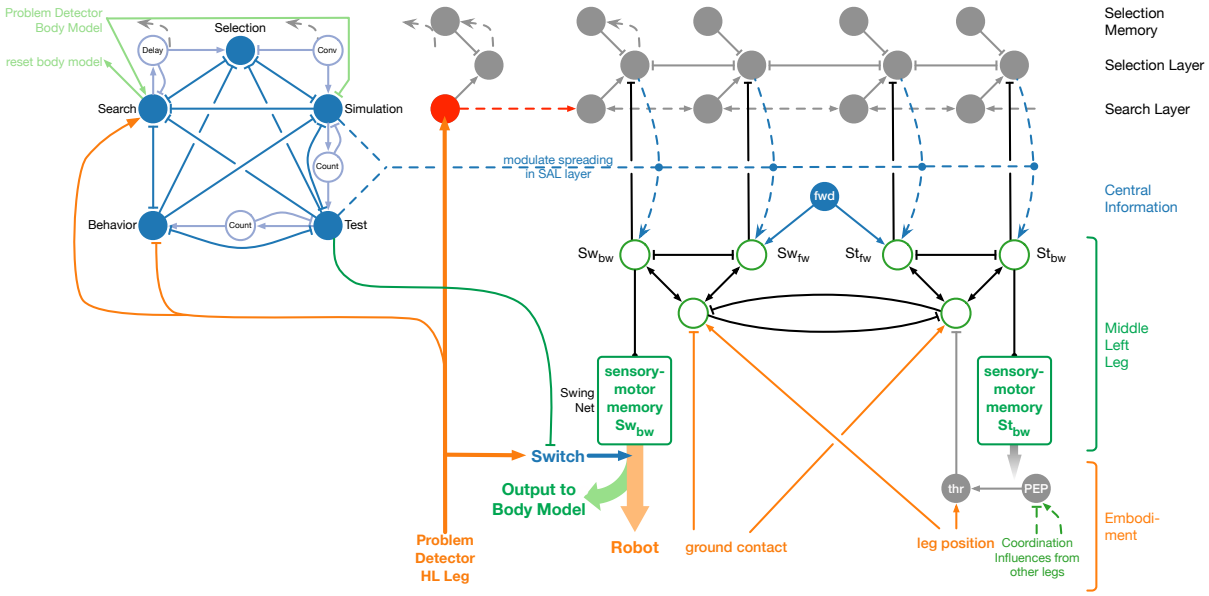

Fig. S7: Detailed leg control network (middle left leg) and cognitive expansion. The blue units (upper left) show the small global network that regulates the state of the whole architecture and represents the different stages. The right section shows a problem detector of the HL (orange), active when an instability is detected, and the controller of the left middle leg in more detail (there is an equivalent controller for each leg and these are interconnected through coordination influences and through local connections between neighboring legs in the top layers). Lower part (green, orange): local controller of ML. Upper part (grey units): three layers that form the local cognitive expansion, which selects an appropriate new sensory-motor memory. The Switch (blue) decides if the motor output is routed to the body model (for predictive internal model/simulation, green arrow) or to the motors to run the robot (orange arrow). Normally, behavior is switched on and the robot is walking. When a problem is detected the network switches towards search (orange, inhibition of behavior and activation of search). This initiates selection of another action: first, information on where the problem occurred is briefly spread in the Search Layer (grey units, top right; in this layer, each search unit that is associated with a specific motor memory motivation unit is connected to two neighbors as shown above; importantly, there are as well connections to behaviors in neighboring legs (dashed gray arrow at the top right) which allows to spread the search to behaviors in legs close by). Second, a single activation is selected in the WTA selection layer (grey units). These units are inhibited by the Selection Memory which remembers which behavior was already tried in internal simulation. After a selection, internal simulation is run for a given time which is regulated by the small state network (shown at the top left, blue unfilled circles represent units that measure timing). When a problem occurs during internal simulation, simulation is inhibited and a new search is initiated (light green arrows are fed back from internal model and affect the current state of the system in the network shown in blue at the top left). After successful internal simulation, it is switched to the stage “Test”: the switch is operated again and commands of the successfully simulated action are now routed to the robot. When successful, normal walking continues (stage “Behavior”). Motivation units for swing back (SWbw), swing forward (SWfw), stance back (STbw), stance forward (STfw). These units are centrally activated by other motivation units (blue, fwd). Other motivation units (SW, ST) are stimulated by sensory input (e.g. ground contact, leg position, and coordination influences) passing a threshold (thr). Embodiment refers to both physical robot (first order embodiment) and body model (second order embodiment). Connection between units: arrows indicate excitatory influences, bars indicate inhibitory influences.

In case a problem is detected, the cognitive expansion will be activated. This, first, leads to an interruption of walking and a selection of a new sensory-motor memory including its motivation unit, which is not used in the current context. This is followed by an internal simulation to test the suitability of the new action selected. If a sensible solution is found, it will be applied to the robot.

In the following this will briefly be illustrated (see Fig. S7, Figs. 2,3, and 4, for details the reader may be referred to Schilling and Cruse, 2017). The control system forming the cognitive expansion has different control states (called stages, shown in left part of the Fig. S7, this constitutes one global sub-network): During walking, the system is in the behavioral stage (Fig. S7, Behavior, lower left blue unit). When a problem is detected, the system switches to the next stage, Search. A “problematic situation” is characterized here as an unstable posture that occurs during walking and that could be detected if the leg still experiences load although it is actively unloaded (for example Fig. 3, beginning of grey space). The problem detector is realized simply as a mathematical calculation of static stability (see section above). If the problem detector is activated, it stops the execution of the behavior. It further activates the switch decoupling the body from the control process (Fig. S7, central lower part, green bolt arrow). Therefore, the behavior causing the problem is stopped. At the same time, the problem detector is activating the search stage for a solution, i.e., it selects a new action out of its current context: this is realized in three sub-stages that are related to three sub-networks (Fig. S7 and S8, Fig. 2). For each of the four motivation units per leg (green units), that represents a sensory-motor memory (Swing forward and backward, Stance forward and backward for each leg), there is a corresponding unit in each of the three added layers (search – selection – selection memory layer, grey units, upper right; for convergence of these networks see Fig. S8). In the “search” layer an activation is induced starting at the problem detector that caused the internal simulation (n the case of our example, trying to lift the left hind leg lead to an instability; this stopped behavior and in the search layer an activation in the Swing_forward unit of the left hind leg is induced). This activation spreads slowly through the layer (modulated by the stage neuron “Search”) and at first activates units close by (most probably activating at first one of the four other behaviors in the same leg, but later-on spreading inside the search network towards neighboring legs). This favors actions that are morphologically close to the cause of the problem. Each problem detector is connected to at least one unit of the search layer, which initiates the spreading of activation.

After some units of the search layer became activated, in a next step one unit shall be selected in the selection layer forming a winner-take-all net. The selection is done in the selection stage (see Fig. S7 upper left) which is automatically triggered after a couple of iterations and ending the search stage. The search and selection layer are connected in a one to one fashion, i.e., for each unit in the search layer there is a corresponding unit in the selection layer. As long as the spreading activation is running, the search units’ activation transfers directly to the corresponding selection unit (again, this connection is modulated by the unit representing the search stage, modulation is not shown). Inside the selection layer, units are connected in a winner-take-all fashion inhibiting each other. As a consequence, activations will converge until only one unit is active inside the selection layer (for details of convergence, see as well Fig. S8). Motivation units of currently active behaviors in the system (lower part, green units) are excluded from the selection through inhibiting connections from the motivation units towards the corresponding units in the selection layer. Similarly, an active “search memory” unit inhibits activation and selection of an action. “Search memory” units become activated once an action was selected and run in internal simulation. This circumvents applying the same action twice and realizes a form of working memory (as assumed required for such processes and already present in insects (Giurfa & Menzel, 2013)).

In the winner-take-all network (selection layer) only the unit with the highest activation stays active and inhibits all other activation after the network has converged for some time (after about 10 to 20 iteration steps). When the WTA has converged, it has selected a single new behavior which is morphologically close to the origin of the problem and should be now tested in internal simulation. After convergence of the selection layer, the network switches to the simulation stage (see Fig. S7, top left). The active unit of the selection layer is activating the corresponding motivation unit (green circle) and in this way initiates the behavior. This activation of a behavior may affect not only the explicitly selected behavior, but may have direct effects on the selection of other behaviors. Crucially, during internal simulation the behavior is not carried out on the robot itself, as the motor commands are rerouted towards the body model instead of the motor system (such a decoupling is assumed in mental simulation in humans (Hesslow, 2002)). Instead it is applied on the internal model (Fig. S7, Fig. 2, green bold arrow) and the model predicts the consequences of the simulation of this behavior. Internal simulation runs for a given time: after search and selection of a behavior, the control structure switches to a “Simulation” stage (shown top left, blue unit “Simulation”). This stage is kept active for a specified time (450 iteration steps, simulating at least two steps of the robot). The stage is, however, aborted when a problem occurs in internal simulation. Importantly, the internal model is equipped with the problem detectors as given in the real agent. Only in this way the internal model can decide if the problem is still present and the search has to be started again by letting the spreading activation start over.

As soon as a sensible solution has been found during simulation, the test stage (Fig. S7, top left and blue unit "Test") is activated and the switch turns back to reroute the output of the sensory-motor memory to the motors of the robot (orange bolt arrows), so physical walking can resume again.

The structure of the network organizing this process is given in Fig. S7, left upper part. Again, we provide the connection matrix for this small network in Table S4 (for details see the open github repository with code implementation –CognitivePhases.py).

The connections between units organizing the global stages of the whole system are sparse and given in Table S4. Upper row and left column refer to the individual motivation units. Weights arranged in a horizontal row mark input weights of the unit shown at the left. In principle, the five units representing the stages (Behavior – Search – Select – Simulation – TestBehavior) are connected through inhibitory connections to units representing stages following each other (see red values of -0.25 between these units) and excitatory signals that switch between states. Units in italics represent units that further show simple temporal behavior which could be understood as a low or high-pass filter, but was simply realized as an internal counter. *StartPlan* is simply a unit which is delaying input and fires five timesteps later. *PD_Delay* is a unit that delays the problem detector signals for one time step. *SimCount* and *TestCount* have phasic characteristics and stay active only for a specific time window of 400 respective 100 timesteps. The two last units trigger additional side effects. *ResetSearch* simply resets all units in the search and selection layer after a successful solution was tested as a behavior. This is necessary as otherwise the network would keep an internal state alive that would influence the next time a problem arises. *NextRun* is triggered when a solution tested in internal simulation was unsuccessful (caused by problem detectors in the internal model). As a consequence, internal simulation is stopped (-2 weight towards *Sim* stage unit) and another search is started (+2 weight towards *Search* stage). Further, the winner-take-all selection mechanism has to be reset as a side effect (which could be implemented as an inhibitory influence towards all those units, but this would be a lot of connections and lead to quite inefficient computation only for a simple reset). For further details see Schilling and Cruse (2017).**Table S4:** Connection weights for the different processing stages regulating switching from behavior towards searching, selecting, and testing an alternative behavior out of context.

	Beh	Search	Select	Sim	Test	*StartPlan*	*PD_Delay*	*SimCount*	*TestCount*	*ResetS.*	*NextRun*	*All PDs*	*BM PDs*	Bias
Beh	1	-0.25			-0.25	-1			2					
Search	-0.25	0.75	-0.25	-2	-2	1	-1							
Select	-4	-0.25	1	-0.25	-4	*1*	1							
Sim	-1		-0.25	1	-1			-2						
Test	-0.25			-0.25	1			0.75	-2					
StartPlan					-1	1						*Activate Cogn. Exp.*		
PD_Delay				-1		1								
SimCount				1										
TestCount					1									
ResetS.									1					
NextRun				2									*Reset Int. Sim.*	-2

In Fig. S8 an example illustrates the dynamics of units forming the three layers (search, selection, memory, as shown in Fig. 3 and S7 top three rows of units) for six sensory motor memories of the cognitive expansion (Fig. S8, left). This network is modulated by the global stage units (blue units, top left in Fig. S7): First, during the search stage activation is spread out in the network (red units on the left, initial activation is induced by problem detector). Second, in the selection stage a single unit is selected in the middle layer (Fig. S8, green units that form a winner-take-all network). Activation is driven by activation of the corresponding units in the search layer. Further, the selection is inhibited by the third unit (selection memory, blue unit on right) that gets activated after an action was selected once. In this way, this layer remembers which action should not be selected a second time. In the search layer (Fig. S8, left, red units) activity spreads only during search stage, in between activity is maintained (but there is some noise). In the selection layer (green units), selection of an action is only considered during selection stage.

This structure consisting of three layers locally expands the Walknet structure. We provide the extended connection matrix for this network in Table S5 showing only the three units for the two swing units (for details see the open github repository with code implementation –SpreadAndSelectNetwork.py). Upper row and left column refer to the individual motivation units. Weights arranged in a horizontal row mark input weights of the unit shown at the left. In principle, the five units representing the stages (Behavior – Search – Select – Simulation – TestBehavior) are connected through inhibitory connections to units representing stages following each other (see red values of -0.25 between these units) and excitatory signals that switch between states. Units

in italics represent units that further show simple temporal behavior which could be understood as a low or high-pass filter, but was simply realized as an internal counter. *StartPlan* is simply a unit which is delaying input and fires five timesteps later.

Table S5: Connection weights for the three additional layers of the cognitive expansion (only shown for *swing_forward* and *swing_backward*). The table corresponds largely to Table S3. New entries are marked in bold. First, activation in the search layer is induced by the problem detectors (*all PDs*). It spreads there to neighboring units (blue 0.3 activation). Blue weights signal that these connections are modulated and only active during the stages introduced above, i.e. the search spreading activation only progresses while the system is in the search stage. The *Search* units activate the corresponding *Select* units which are forming a simple WTA network from which only a single unit will emerge active after some time. This process is further influenced by the *Memory* units that memorizes alternative behaviors which have been already tried and failed in internal simulation.

	fw	bw	leg_MU	sw	sw_f	**Search**	**Select**	**Memory**	sw_b	**Search**	**Select**	**Memory**	st	*All PDs*	st_f	st_b	*beh_PEP*	*GC*	sw_start
fw	*Driven by global input*													*Stop Walking and activate Cogn. Exp. + active problem detector induces activity into Search units.*					
bw																			
leg_MU			rec_w																
sw			0.25	rec_w	0.8				0.8				-0.75				6.75		6.75
sw_f	-0.15			0.45	rec_w		**2**		-0.6										
Search						**1**				**0.3**									
Select						**0.5**	**1**	**-0.2**			**inh**								
Memory							**1**	**1**											
sw_b		-0.15		0.45	-0.6				rec_w		**2**								
Search						**0.3**				**1**									
Select							**inh**			**0.5**	**1**	**-0.2**							
Memory											**1**	**1**							
st			0.25	-0.75									0.25		0.8	0.8	-6.75	2.25	-6.75
st_f	-0.15												0.45		rec_w	-0.6			
st_b		-0.15											0.45		-0.6	rec_w			
sw_start				1															

Figure S8 shows how activation progresses in these three layers of the network: The right section illustrates the activation of the three units of each of the six memory elements depicted here (search: red line, selection: green line). At the beginning, memory element 2 (from above) is activated by the problem detector (search on, but no selection). Via spreading activation the neighboring units (element 1, 3, to a lesser extent 4 – 6) are activated. Due to WTA properties, unit 1 (red) wins and search is activated (green). This is apparently not successful, which elicits a further spreading activation, which now activates unit 4 (red, note that earlier units are inhibited via activation of selection memory (not shown). As a consequence, selection is activated (green). This apparently is not successful, too, which leads to activation of unit 5 (green).

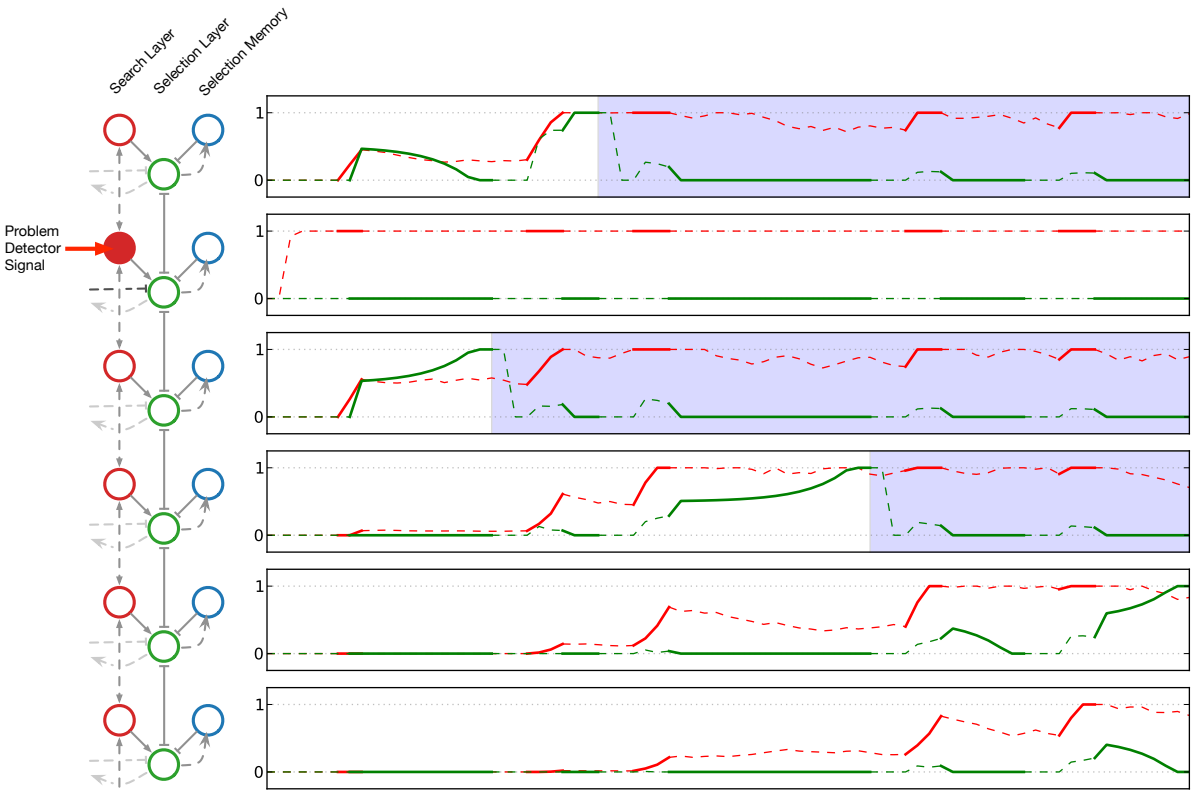

Fig. S8: Convergence of search and selection process. Illustration of the local part of the cognitive expansion that consists of three units for each behavior (sensory-motor memory) of the control network (Fig. S7, grey units, upper right). On the right, the corresponding temporal activations (between 0 and 1) of the network are shown during a run of the cognitive expansion (time from left to right). Red lines show activity of search unit, green of the selection unit). Solid parts of line indicate activity during the respective stage, dashed lines show activity of the network that does not affect the behavior of the system. Blue background area indicates activity of the selection memory unit after the specific behavior (shown in one row) has been selected. This inhibits the subsequent reselection (shown in green) of this action.