Communicating with Executable Action Representations

Malte Schilling[1,2] and **Srini Narayanan**[2]
[1] Center of Excellence Cognitive Interaction Technology, Bielefeld, University
[2] International Computer Science Institute Berkeley, 1947 Center Street, Suite 600, Berkeley (CA) 94704 USA
Email: mschilli@icsi.berkeley.edu

Abstract

Natural language instructions are often underspecified and imprecise which makes them hard to understand for an artificial agent. In this article we present a system of connected knowledge representations that is used to control a robot through instructions. As actions are a key component of instructions and the robot's behavior the representation of action is central in our approach. First, the system consists of a conceptual schema representation which provides a parameter interface for action. Second, we present an intermediate representation of the temporal structure of action and show how this generic action structure can be mapped to detailed action controllers as well as language.

Introduction

We use communication as a tool in order to convey a meaning to another person. Often we communicate with somebody explicitly to interact with him. Cooperation appears to be one root of communcation (Tomasello 2008). But it presupposes several requirements for a successful communication. Both communicating partners want to achieve something together. And they rely on a set of shared knowledge.

Modeling approaches have tried to capture such shared representations and have tried to build human like representations. But this turned out to be quite difficult. Early-on traditional AI approaches focussed on high level and merely symbolic representations. This has shown to be insufficient and lots of information is missing. Language input is only providing parts of the used information and is often underspecified. What is needed is, first, the connection of the higher level information to lower level representations, i.e. a grounding of the representation. Second, the integration of additional knowledge is necessary. Information about the current situation, the ongoing context of the interaction, afforded goals, possible actions or simply ontological knowledge about relations between objects has to be incorporated.

Following a bottom-up approach, we start from a simple lower level representation of sensorimotor behavior. While such representations seem quite different from linguistic representations, there is broad support from neuroscience and behavioral science showing that in the human brain language, communication and cognition are rooted in the same representations (Jeannerod 2006). The structure of linguistic representation reflects the structure of the underlying conceptual system (Narayanan 1997) and the structure of actions shows up in the way how we talk about actions and how we think about actions (Pulvermüller et al. 2005).

Central here is first, the idea of recruitment (Anderson 2010) and, second, the notion of internal simulation (Hesslow 2002): Grounded internal models are flexibly utilized in service for cognitive tasks. These models originally served a specific behavior and co-evolved in a different context (Steels 2003). But the resulting multimodal and predictive models can be flexibly used in different contexts decoupled from the original task. Planning ahead becomes a form of internal simulation, i.e. trying out alterations of existing behaviors in an internal simulation with the body decoupled from the control system.

In the Neural Theory of Language (Feldman and Narayanan 2004) language understanding is explicitly understood as invoking such an internal simulation. The internal simulation is the process in which meaning of language is unfolded. The language input drives the internal simulation by activating at first the linguistic representation. Activation spreads to the connected different types of knowledge representations and these become part of the internal simulation. In the internal simulation all the missing details of the underspecified linguistic input are available as they are part of related representations and are pulled into the internal simulation. A key aspect of an internal simulation is the temporal unfolding of events. And the temporal structure of an internal simulation is organized by the structure of actions.

While internal simulation provides a process for integration of knowledge, the gap between representations on a linguistic level and as used in motor control seems to be quite big and not easy to overcome. Narayanan proposed Petri-network based schemas as a representation of actions (Narayanan 1999). First, he identified a general structure of action which is also reflected in language. Second, in a specific type of Petri-networks he provided a mechanism which can represent a temporal organization of actions and events. This representation is at the same time executable. In this paper, we want to follow up on this type of representation of action as an intermediate representation between high level representation and low level process models of behavior. We will explain the x-schema representation and its features and

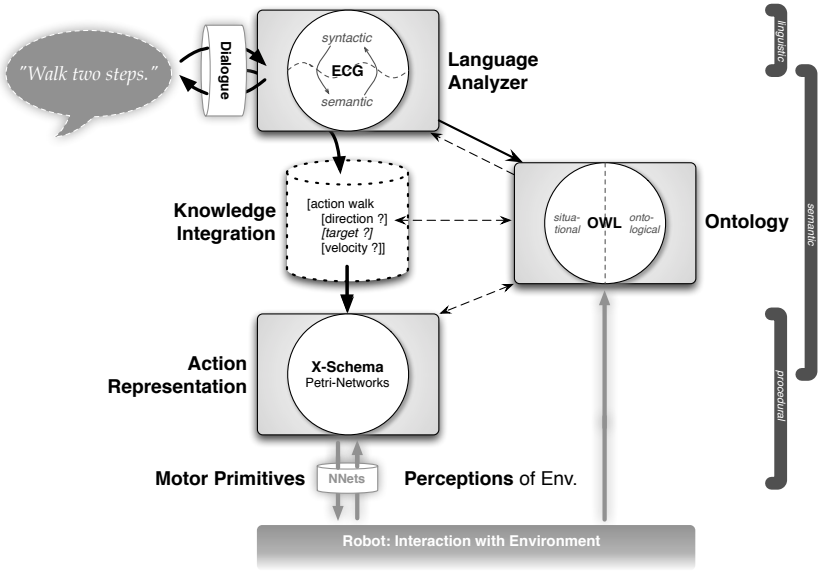

Figure 1: Schematic overview of the representation system: Sensorimotor circuits (bottom-left) are directly linked to the x-schema representation. Language is represented through Embodied Construction Grammar (upper-left) in which syntactic and semantic information are tightly connected. All knowledge representations are connected to an ontology that defines concepts, their parameters and relations (right). Language instructions are translated into constructions and the resulting semantic specification is complemented (middle, knowledge integration) with the missing information in order to initiate behaviors and setup sequences of actions.

will show how this can be directly mapped to the linguistic representation and to sensorimotor circuits. We apply this system in a human-robot interaction in which a robot shall be guided through language input by a human user. We provide a simple example of a six-legged simulated robot moving around in an unknown environment. As we are still on the beginning of this work, we will discuss the benefits of such an approach of connected representations and how the interconnected representations could be exploited to generate meaningful answers. In addition, we want to discuss the extension to more challenging domains.

Knowledge Representation

Different forms of representation are suited to represent the different kinds of knowledge inside the system (see fig. 1). Starting on the lowest level with procedural knowledge, forming upwards an ontology of concepts and on the highest level linguistic representations containing words and syntactic information. Each level demands its own specific set of requirements and asks for a specific formalism. In the following, we will briefly explain the different representation formalisms, first, on the lowest level and, second, on the linguistic level. This will emphasize the large gap between those two types of representation and will motivate the introduction of the intermediate action structure representation.

Sensorimotor Level: Neural Networks

We deal with a control system for a six-legged walker. The basic system is based on work (Schilling, Cruse, and Arena 2007) on the walking of the stick insect. The behavioral repertoire is quite small. Basically, the animal can move around in an environment. A movement consists of a velocity and a target. Each leg is individually controlled and can perform two basal movements: A swing or a stance movement. These movements are realized as simple neural networks (fig. 2) and they have additional parameters, e.g. the stance movement requires a step width.

Even though the control system is quite simple, it produces stable and robust behavior as it can deal with severe

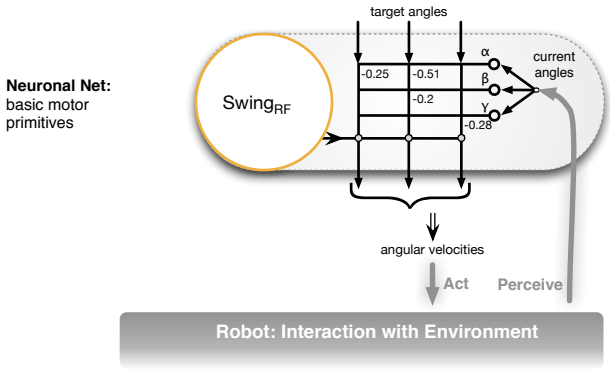

Figure 2: Procedural level: Neuronal Networks control the robot/agent and get perceptions from the robot in the dynamic simulation environment. The overall behavior is organized on top of that in Petri-Networks that control the switching and timing of the motor primitives.

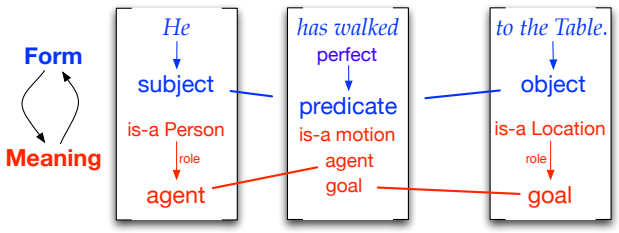

Figure 3: Example constructions. A construction consists of a form part (blue) and a meaning part (red). It is defined by parameters and by relations or roles that can connect to other constructions (e.g., the agent part of the *walk* construction relates to the subject of the sentence which fills this role.)

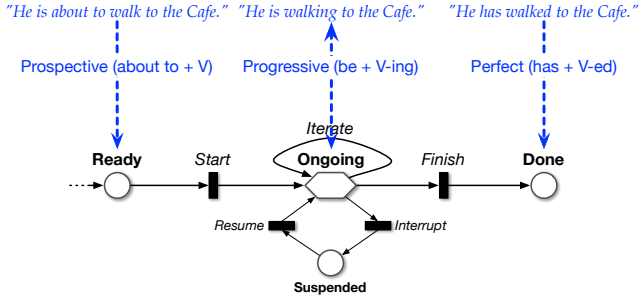

Figure 4: Example of different verbal aspects. While the general action structure for all sentences is the same, the three different sentences map to different activations (markings) of the same Petri-network representing the action.

disturbances, e.g. the loss of legs. This is one of the advantages of such systems: they are embodied and situated and the overall behavior emerges in the continuous interaction with the environment. From a representational perspective it is crucial that the internal states are directly linked to the behaviors themselves and are grounded in the sensory experiences. One example of a more complex grounded internal representation is a model of the own body which can be used to control the movements of the legs during the stance phase. Such an internal model can be used for planning ahead through internal simulation. In this way the model can overcome novel challenges like crossing large gaps.

Language Level: Constructions

Language is modeled using Embodied Construction Grammar (Bergen and Chang 2005). Constructions are form-meaning pairings that link word forms and conceptual schemas. On the one hand, one part of a construction covers the form. These can be morphemes, words or complete idioms. In addition, the form of a construction can establish rules how constructions can be combined. This allows to express complex syntactic information through constructions.

On the other hand, constructions express semantics. The meaning part is a conceptual schema representation which allows to define relationships and parameters. Meaning and form part are tightly intertwined. In particular, when constructions are connected, such a connection is not restricted to the form or meaning part of the involved constructions. Instead, a construction defining the connection determines how meaning and form interact, e.g. Fig. 3. The construction for the verb *"walk"* requires on the meaning side an agent (who is walking) and as a parameter a target location (where to go). This structure can be mapped to different sentences, e.g. to the one used above. This sentence is derived from a construction that generates the typical syntactic structure of *subject–predicate–object*. As a part of this the subject of the sentence is bound to be the agent in the situation. This construction defines the form-meaning pairing for such active sentences. At the same time, the agent is restricted to a specific concept. An agent has to be of type person. In the same way, the target of the movement is connected to the object of the sentence.

Representation of Action: X-Schema

There is a broad gap between sensorimotor circuits and linguistic representations of actions. An action is defined, first, by its interface, i.e. a set of parameters of the action. An action consists of parameters which can be either variable or can be restricted to a certain value, e.g. velocity is a parameter of a movement (*walk*) and for a fast movement the velocity is high (*run*). On the language side, constructions provide such a representation of meaning. The semantics are represented as conceptual schemas. This parametrization is shared by the linguistic representation and the actual action controller. The conceptual structure is also expressed in our system in an ontology which acts as a repository of possible concepts and reflects parameters and roles of concepts. A representation of the current situation, the state of the system and current percepts, is connected to this ontology (This is not explained in detail here. For the simple case of the hexapod walker sensory states are restricted to proprioceptive feedback and there is only a very simple map-like representation of the environment for navigation.). The ontology provides a mapping and default values for the actual controller of the action, e.g. it establishes the actual value of velocity for the linguistic parameter *fast* in the walking controller.

Second, crucial to an action is its temporal structure. On the one hand, the handling of the temporal unfolding of an action is the basic task of sensorimotor representations. Dynamical system approaches are well suited to express complex temporal and spatial relations in multiple dependent or independent dimensions. On the other hand, there is a general temporal structure of actions and events which consists of a common state space for action, e.g. an action can be ongoing or finished. This generic temporal structure of an action is also reflected in language. For many languages a specific grammaticalization has developed to denote the state of an action. This is called verbal aspect (Fig. 4). In order to be able to reason about events, we need a state space which includes this set of states, but in addition also the possibility to describe changes of states. We use x-schema to represent this temporal structure of actions. X-schemas are an extension of Generalized Stochastic Petri Nets (Mu-

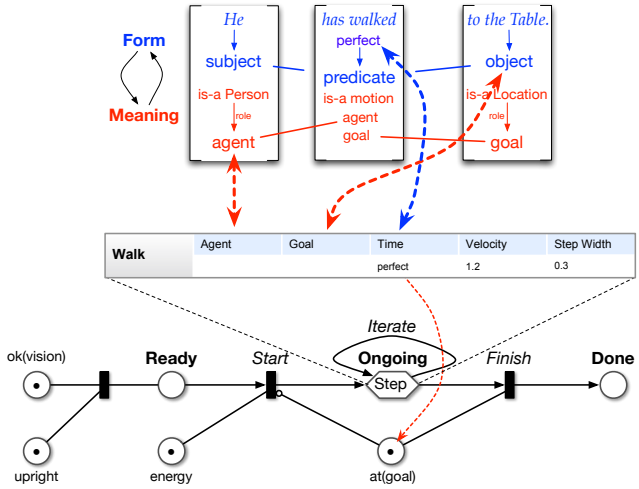

Figure 5: Showing the connection from linguistic representation (upper part, constructions) to an x-schema. The x-schema is defined by the parameter interface (in the middle). The Petri-network (lower part) consists of a generic set of action states (Ready, Ongoing, Done) and describes the temporal structure of the process. The hexagonal step represents an embedded x-schema in which the motor primitives are directly controlled.

rata 1989). Petri-networks are graphs consisting of places ($\mathcal{P}$) and transitions ($\mathcal{T}$). Places can store resources and in this way a marking of the graph describes a state of the system. Transitions describe how markings change in the graph over time. There are different types of transitions allowing to model the movement of token between places. Special transitions can, for example, create token, inhibit the creation of a token, consume tokens, wait before they fire or can distribute tokens in a stochastic fashion (for details (Narayanan 1999)). Petri-nets are well suited to describe temporal processes as they allow concurrent processes and can handle synchronization of these. In addition they are able to represent resources and the consumption of these.

An x-schema consists of a Petri-net describing the unfolding of an action. It consists of places representing the lower level motor primitives. The activation of these places is directly coupled to the activation of the motor primitives. But in extension to a normal Petri-network an x-schema is, on the one hand, connected to an interface that provides access to the parameters of the action. On the other hand, the general action structure is part of the Petri-network which means that for each generic action state there is a corresponding place in the x-schema (Fig. 5). This allows, first, the direct mapping of an x-schema to language (Fig. 4). Language can parametrize directly an action. And it expresses the verbal aspect of the action. Secondly, an x-schema provides a simulation semantics. It is directly grounded in the sensorimotor control as single states correspond to basal motor primitives (e.g. a swing controller) and sensory experiences. The activation of the motor primitives is organized by the x-schema graph. This level of representation allows to combine different Petri-networks (subnetworks are shown in the figures as hexagonal states) and to reason about actions by internally running the networks. Following the activation traces in the x-schema in such an internal simulation directly leads to the activation of predicted sensory consequences or invokes dependent procedures. As a consequence, x-schema are ideal for planning ahead and synthesizing complex behavioral plans. While they are abstract enough to provide a small space for planning, they still directly link language and action processing, as x-schema can handle temporal structure and can be mapped to conceptual semantic schemas.

Knowledge Integration

Language input is usually underspecified. Parts of an uttered sentence connect to background knowledge or what has been said before. A central process is therefore the integration of knowledge (Fig. 1). The main task for the knowledge integration is to decide which action to take and to provide all required information. Here this is directly driven through the given language instruction. In the simplest case the knowledge integration passes the action command from the language processing to the action execution. In more complex cases it has to infer or has to look up missing information. For example, when an action should be invoked which preconditions are not met, this process has to come up with a behavioral sequence of actions that allows to execute the specified action. Searching a sequence of behaviors has shown to be quite difficult. But the level of x-schemas provides a good level to plan such behavioral sequences. In an x-schema pre- and post-conditions are explicitly stated while the exact details of the execution of the action are hidden away.

Robot Command Scenario & Future Work

We want to explain how the different representations contribute to language understanding. First, we present the simulation of a six-legged walking robot. Second, we will explain ongoing work connecting our system to ROS (Robot Operating System) and to the control of the PR2 robot.

Instructions for a Hexapod Robot

The six-legged robot Hector is realized in a dynamical simulation environment (Fig. 6, the robot is currently under development at Bielefeld University (Paskarbeit et al. 2010)) and can be controlled using simple command sentences. An uttered instruction invokes an action representation which should be carried out by the robot. The knowledge integration takes care of gathering all required information and initiates the execution of the action.

In the easiest example, all information is given in the language input, e.g. the robot should *"walk slowly two meters to the North"*. A walk action consists of an interface requiring values for a velocity and a specification of where to go.

But usually not all information is given and instructions are incomplete: *"walk two meters to the North"*. The specification of parameters for the action is incomplete and the knowledge integration process has to fill in the missing details, e.g. it looks up a default walking speed which is used to execute the walking.

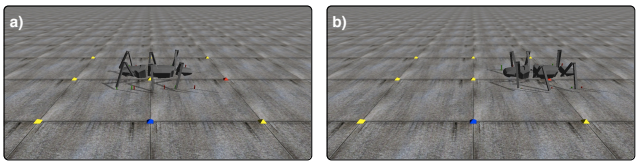

Figure 6: Simulated six-legged robot walking forward.

A typical command could be even shorter or the information could be incompatible to the structure of the action schema, e.g. one could advise the robot to *"walk to the table"*. A default velocity could be looked up in the ontology, but the target is not given as a distance and direction. While it is possible to calculate distance and direction from the known positions, this is not always the case and not cognitively plausible. We do not decide beforehand how far and in which direction to go, as small errors might accumulate during execution. Instead, we would head into the direction until we reach the table. And we can handle instructions which can't be evaluated completely before the execution, e.g. *"walk North until you see the staircase to your left."* This presupposes that information can be integrated into the action during execution. A simple prespecification can't handle such instructions. But the proposed x-schemas provide a mechanism. In the mentioned examples, a target parameter is part of the walk action interface. This parameter is not a single value, but basically the network contains a place which actively determines if the target is reached.

Already slightly more complex instructions make use of the temporal structure of actions. When chains of actions are described or the execution of actions shall be altered this has to be reflected in the way the actions get activated. The example sentence *"walk North until you see the staircase"* could be rephrased as *"stop walking North when you see the staircase"*. There is no explicit initiation of the walking behavior, but to reach the Finished-state of the walking behavior this behavior has to be activated first. Without exploiting the temporal structure of actions a planner would only use the goal information to come up with a suitable behavior leading towards the goal (being in a position from which the staircase can be seen). This might lead to testing out in internal simulations a large repertoire of unhelpful behaviors. Instead in our case the knowledge integration gets as input a desired state of the walk x-schema. First, the goal condition has to be met for the robot. And second, the Petri-net shall move to the Finished/Done state. From this it is quite easy to infer that only the walking x-schema has to be activated.

Interacting with a Robot

The command scenario is only a first step. The advantage of using a generic action structure shows when we turn to more complex tasks and descriptions of entire sequences of action including temporal relations as such instructions can be directly synthesized into chains of actions.

The main advantage of the interconnected representations is that execution and planning processes are directly linked to language understanding and both processes can mutually inform each other. This becomes apparent especially when something goes wrong, e.g. the execution of an action brakes down or information is missing during execution. As the state of the lower level is directly linked to the higher level of representation it can be used to guide the search for an answer on all levels. For example, walking consists of an iteration of swing and stance movements. A swing movement is finished when a leg touches ground. But when the robot stands in front of a gap the leg can't find a foothold and the stance movement can't be initiated. The execution will be stopped. While the lower level can not solve this problem, a user might give advise, e.g. to move into a different direction or, for a small gap, to change the step width. The robot could be instructed to make a small step to the border of the gap and then cross the gap with a large step.

While the system can integrate the new instruction, it can also give feedback on its current state: It can tell that the swing behavior aborted because there was no ground contact found. There is currently no natural language output. Only state information can be returned to a user. In the future we are going to extend the system and want to give informed language feedback. Here the generic structure of the action allows to directly give information about ongoing processes and provide details about the process execution. The hierarchical action representation can be used to address in which stage the execution run into problems and which conditions were not met. In the long run, the system should be extended towards dialogue and continuous interaction with the user. On the one hand, being able to give feedback. On the other hand, integrating new language input continuously during the execution and possibly guiding planning processes.

The walking scenario is quite simplistic and currently we are applying the system to a more complex manipulation scenario. We are connecting the knowledge representations and the knowledge integration to ROS and want to use it to control the PR2 robot in a manipulation task. In the task the system will be extended to more complex action, e.g. grasping an object. An action like grasping is much more demanding as the simple example action representations addressed in this paper. The presented Petri-net representations consist only of sequences of action. But during grasping there are multiple concurrent processes: while a reaching movement is carried out, the hand or gripper is preshaped to fit the object. Usually, such complex temporal relations are encapsulated in a movement controller. But it is one of the strengths of a Petri-net representation to be able to handle concurrent process flow. One advantage of our system is that this level of detailed temporal process descriptions and interrelations of multiple processes is directly linked to language. It is our goal in the future to synthesize complex actions which are not simple sequences of action, but can have a rich temporal structure. A more complex structure and especially rich temporal relations require that behavioral plans can not be given in advance. Instead, it can become necessary to construct new Petri networks during runtime from basic building blocks and alter the single action units. Such synthesized action shall be generated from language instructions. This requires to fill in missing information at runtime or to complete behavioral plans and fill in necessary missing actions.

Discussion & Related Work

There is a long standing tradition of work on natural language instructions going back to Winograd's work on SHRDLU (Winograd 1971) in which simple actions were applied in a simple virtual environment. This has been extended to work in robots which can be instructed using natural language leading to a wide range of different architectures that can handle language of different complexity. Starting from simple keyword triggered actions up to approaches that can deal with grammar. The goal in many of these approaches is processing natural language into a behavioral sequence. For example, Dzifcak et al. (2009) translate an instruction given through natural language into a formal logic description of a goal which is then used to generate an action script fulfilling the goal. More and more approaches are incorporating semantic specifications and connecting the language to conceptual schema representations, a typical example is the representation of spatial descriptions (Kollar et al. 2010).

While many of these approaches are only taking into account simple sentence structure, Tellex et al. (2011) is extending the work of Kollar et al. (2010) using more complex input which includes hierarchical structure of language in a robot navigation task. In their work parts of the language input are tried to be mapped to objects, places or schematic descriptions, e.g. of a path. The results are probabilistic graphical models describing spatial features. From these a suitable one can be used as a parameter for an action.

Our language processing using Construction Grammar allows to handle complex hierarchical organized language input. Syntax and semantic are processed concurrently and are tightly interconnected. The result is a semantic specification which is related to conceptual schema definitions.

One feature differentiating our approach from others is the intermediate action representation. Similar to some of those mentioned above this builds on the schematic structure of actions assuming a given parameter interface for an action. But in addition, there is a general temporal structure as it is reflected in verbal aspect in language. This more fine grained representation of temporal relation can be transferred to the level of motor control and in this way language can be used to setup quite complex action plans. The connected representations allow to directly link the action structure to language. On the one hand, complex instructions about concurrent events or the sequencing of actions can be translated into behavioral plans. On the other hand, in the future this type of information can be used to produce better user feedback and to generate language output giving detailed information on the course of action or possible problems in the execution.

Acknowledgments

This work has been supported by the Center of Excellence Cognitive Interaction Technology (EXC 277) and by a DAAD postdoctoral fellowship.

References

Anderson, M. 2010. Neural reuse: A fundamental organizational principle of the brain. *Behav. Brain Sci.* 33:254–313.

Bergen, B., and Chang, N. 2005. Embodied construction grammar in simulation-based language understanding. In Östman, J.-O., and Fried, M., eds., *Construction Grammar(s): Cognitive and Cross-Language Dimensions*. John Benjamins, Amsterdam.

Dzifcak, J.; Scheutz, M.; Baral, C.; and Schermerhorn, P. W. 2009. What to do and how to do it: Translating natural language directives into temporal and dynamic logic representation for goal management and action execution. In *IEEE Int'l Conf. on Robotics and Automation (ICRA)*, 4163–4168.

Feldman, J., and Narayanan, S. 2004. Embodied meaning in a neural theory of language. *Brain Lang.* 89(2):385–392.

Hesslow, G. 2002. Conscious thought as simulation of behaviour and perception. *Trends in Cognitive Sciences* 6(6):242–247.

Jeannerod, M. 2006. *Motor Cognition — What Action tells the Self*. University Press, Oxford.

Kollar, T.; Tellex, S.; Roy, D.; and Roy, N. 2010. Toward understanding natural language directions. In *Proc. of the 5th ACM/IEEE international conference on Human-robot interaction*, HRI '10, 259–266. NJ, USA: IEEE Press.

Murata, T. 1989. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE* 77(4):541–580.

Narayanan, S. 1997. Talking the talk *is* like walking the walk: A computational model of verbal aspect. In *COGSCI-97*, 548–553.

Narayanan, S. 1999. Reasoning about actions in narrative understanding. In *Proc. Int'l Joint Conference on Artifical Intelligence*, 350–357. Morgan Kaufmann Press.

Paskarbeit, J.; Schmitz, J.; Schilling, M.; and Schneider, A. 2010. Layout and construction of a hexapod robot with increased mobility. In *Proc. IEEE RAS/EMBS Int'l Conf. on Biomedical Robotics and Biomechatronics*, 621–625.

Pulvermüller, F.; Hauk, O.; Nikulin, V. V.; and Ilmoniemi, R. J. 2005. Functional links between motor and language systems. *European Journal of Neuroscience* 21(3):793–797.

Schilling, M.; Cruse, H.; and Arena, P. 2007. Hexapod Walking: an expansion to Walknet dealing with leg amputations and force oscillations. *Biol. Cybern.* 96(3):323–340.

Steels, L. 2003. Intelligence with representation. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences* 361(1811):2381–2395.

Tellex, S.; Kollar, T.; Dickerson, S.; Walter, M. R.; Banerjee, A. G.; Teller, S. J.; and Roy, N. 2011. Understanding natural language commands for robotic navigation and mobile manipulation. In *Proc. of the National Conf. on Artificial Intelligence (AAAI)*. AAAI Press.

Tomasello, M. 2008. *The Origins of Human Communication*. MIT Press.

Winograd, T. 1971. Procedures as a representation for data in a computer program for understanding natural language. Technical report, MIT AI Technical Report 235.