Dynamic Symbol Grounding, State Construction and the Problem of Teleology

Erich Prem
erich@ai.univie.ac.at
The Austrian Research Institut for Artificial Intelligence*
Schottengasse 3, A-1010 Wien, Austria
Tel. +43 1 5336112, FAX +43 1 5320652

January 1995

Abstract

Symbol grounding has originated within the connectionist-symbolic debate so as to gap the bridge between the two approaches. This paper provides an overview about recent results concerning symbol grounding, which is critically reviewed here. A thorough analysis reveals that symbol grounding parallels transcendental logic and is best viewed as automated model construction. If this diagnosis is true, the necessary next question must be which sort of models are generated in symbol grounding systems. The answer to this question very much depends on the kind of network architecture employed for grounding. An illustrative neural network architecture is used to explain how a dynamic symbol grounding system generates a formal model. It is argued that, depending on the architecture, very different kinds of signs ranging from input to goal descriptions can be grounded.

1 Symbol Grounding

1.1 The Symbol Grounding Problem

Based on a fundamental criticism of symbolic models and Searles "Chinese room" argument [Searle 80], Stevan Harnad introduced "The Symbol Grounding Problem" in his 1990 paper. Symbol grounding (SG) tries to answer the question as to how it is possible for a computer program to use symbols which are not arbitrarily interpretable. Whereas the meaning of signs in conventional programs is just "parasitic on the meaning in our heads", grounded symbols should possess at least some "intrinsic meaning" [Harnad 90]. The problem can also be formulated as how it is that formal symbol systems can acquire a semantics which is not based on other symbol structures but on the system's own sensory experience.

It should be noted here that Searle—as opposed to Harnad—is more concerned with *intentionality* than with *meaning.* However, as of today, it is quite obvious that symbol grounding cannot solve the problem of original intentionality. The reason for this lies in the observation that intentionality is connected to consciousness, cf. [Frixione & Spinelli 92]. But Harnads proposal is obviously directed towards the generation of correlational semantics. (For the more philosophical discussion see [Christiansen & Chater 92, Christiansen & Chater 93]. Harnads answers can be found in [Harnad 94].)

Consequently, symbol grounding has mainly been performed in the context of linguistic research so as to clarify the question of semantics or the origin of reference in words ("word meaning"). The general idea is to equip a symbol system with some kind of measurement device. The goal of the system is to find invariances in the sensory data whenever a specific symbol is shown to the system. A perfect symbol grounding system would then be able to use a specific symbol in order to describe the sensory input.

Such a system would, of course, also contribute to gap the bridge between symbolic and subsymbolic models of cognition, since it would use symbols for communication but, on the other hand, possess rich semantic meaning for these symbols which would not consist in other symbols, cf. [Dorffner & Prem 93].

*The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry of Science an Research.

1.2 Symbol Grounding Systems

A group of connectionist proposals for network architectures addresses the problem of grounding object categories in perception. Typically, the models possess two types of input: one for (simulated) sensory data, another one for symbolic descriptions of the data. The networks are usually chosen so that the sensory input is categorized trhough un- or self-supervised categorization algorithms like Kohonen networks. The resulting category representations are then associated with some sort of symbolic description of the static input. Such architectures have been suggested by [Chauvin 89, Cottrell et al. 90, Schyns 91, Dorffner 89]. A few models have been suggested which try to ground symbols in dynamic input data, e.g. [Cottrell et al. 90, Nenov & Dyer 94].

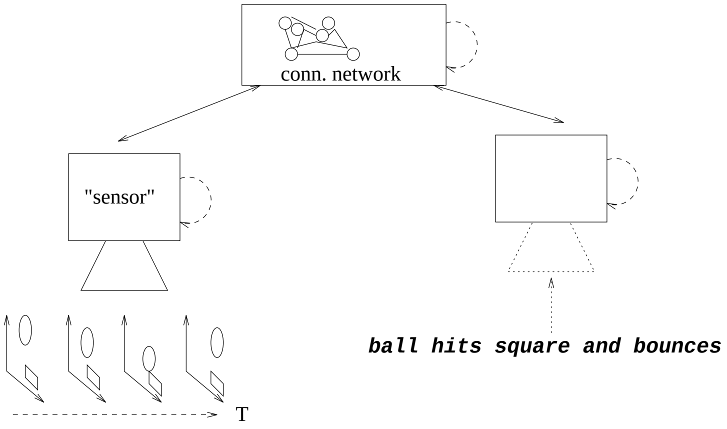

Figure 1: (Simulated) sensory data from visual scences is mapped on a symbol space by means of (recurrent) connectionist methods.

Figure 1 shows the "generic" SG-architecture (generalized to sequences of input). In most of these models categorization of input and naming of these categories is clearly separated in two parts, i.e. in two neural network modules. The motivation for this modularization either consists in assumptions concerning cognitive aspects of categorization (e.g. that children form categories without always having names for the categories [Schyns 91]) or in aspects concerning features of typical AI symbols (e.g. in the observation that "symbols are arbitrary" [Dorffner 89]).

Techniques for generating the categories include Kohonen nets, auto-associative backpropagation networks, and "winner-take-all" strategies. In systems with dynamic input data recurrent auto-associative backpropagation networks are used to find compressed category representations in the hidden layer [Cottrell et al. 90]. The second module connects categories to labels (symbols) and must therefore be trained with a supervised algorithm, e.g. backpropagation.

In the next section I shall explain why and how SG models do a bit more than just labeling inputs.

2 Symbol Grounding and Model Construction

There are three main recent results concerning symbol grounding, which fundamentally change our view of what SG is and can be used for. These results are: (i) SG is, essentially, automated model construction; (ii) SG implements aspects of transcendental logic; (iii) dynamic SG can generate state transition models.

2.1 SG is automated model construction

In SG models the only purpose of symbols is their reference to external objects, respectively to the sensory data generated by them. A theory of SG which largely adopts such a conception (basically a form of correlational semantics) can only result in a specifically scientific model (in the sense of Fig. 2) which is designed to reflect nature for epistemic purposes. This process is equivalent to the automated development of a formal model of a natural system. ("Natural" because we are still talking about *grounded* systems, which are connected to the world through measurement devices.)

This interpretation of SG is not changed by the observation that SG models construct some kind of *subjective* model of their environment—based on the inherently statistical nature of neural network algorithms, because the system itself still generates a symbolic model of this environment. SG systems try to bring a manifoldness

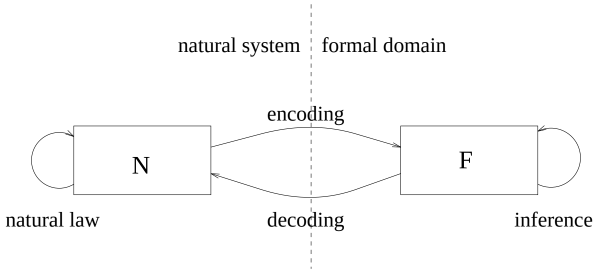

Figure 2: Model of a natural system.

of sensory data into the unity of concepts and *express* these concepts by means of arbitrarily chosen forms. It is essentially this feature which can be found in any mathematical model of a natural system: Some kind of encoding procedure maps “objects” of the environment (usually through employing meters) onto symbols in a formalism. Implications in the formalism are then used to predict what natural law does to the natural system. This situtation is depicted in Figure 2. SG systems try to map sensory data from N on the symbol in F such that this symbol is understood (i.e. used, arbitrarily chosen) by a human teacher.

2.2 SG implements transcendental locic

I have elsewhere argued in detail [Prem 94b, Prem 95a] that the grounding of signs (symbol, index, icon) in SG systems exhibits strong similarites with the three classical forms of reasoning (de-, in-, abduction). SG searches for the conditions which enable signs (specifically symbols) to refer to objects. Therefore it is very close to Kants program of transcendental logic. It can be argued that SG systems are not only grounding *symbols,* but—depending on the concrete architecture—sometimes ground *icons* in the Peircean sense (due to the continuous character of connectionist mappings) and also exhibit features of *indexes* because they are working according to physical laws. (I.e. in the same way as a thermometer signifies temperature could grounded symbols signify what they stand for.)

Let us now concentrate on how the sign becomes connected to its object. In the case of an icon some similarity of representation and represented negotiates between sign and significatum. This reference to a common property takes exactly the form of a logical inference, namely abduction. (M is P. Q is P. → M is Q.) The position of P, the middle term, corresponds to Aristotle’s second figure. Reference of an index to what it stands for is made possible through natural law. Therefore, it is the objects themselves which inform about the set of things referred to. Viewed logically, this process is similar to induction, which informs about the total set of objects with a specific property.

Finally, the symbol is a general and arbitrarily chosen representation of an object. Therefore, it is *the sign alone* which negotiates its meaning. In other terms, the symbol represents in- and extension, it alone ensures that one is connected to the other. This perfect double reference to the singular (extension) and the general (intension) is what makes symbolic negotiation similar to deduction. Fig. 3 tries to support these similarites by showing hierarchical concept trees and how different types of signs relate to the objects they stand for.

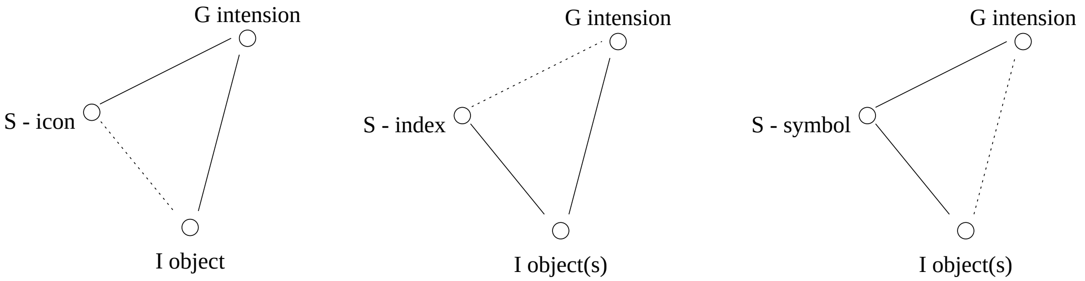

Figure 3: Different views of hierarchical concept trees (arbor porphyriana) and corresponding types of signs. Straight lines show relations given, dottet lines represent conclusions drawn.

In Kantian terms, it are the three logical principles of *sameness* (A is B), *separation* (if B then NON-B), and *opponentship* (A is either B or NON-B, law of the exluded middle) which are realized in SG systems. This observation will allow us to extend this analysis of static SG to dynamic SG. In static ones, states (objects) of the environment are only formalized into symbols. Synamic SG systems should also generate predictive rules. The extension of this discussion to space and time is quite natural and has also been performed by Kant and his successors.

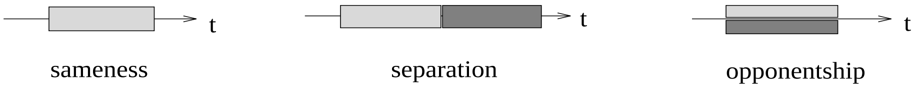

Figure 4: Schematic representation of three transcendental principles.

Fig. 4 tries to appeal to intuition and to suggest as to how the three logical principles can be extended to space and time. *Sameness* creates the permanency of objects in time, *separation* serves as the (inductive) source of creating causal relations (Kant: "The real whereupon something follows."). *Opponentship* puts the "objects" into relation with each other.

2.3 Dynamic SG can generate state transition models

In order to show that SG really is automated model construction it must now be demonstrated how SG systems can generate state transition rules (in addition to state formation). We do this on the basis of the principles of transcendental logic outlined above. Our idea has it been to identify sameness with the notion of a state in a formal model, separation with the timely ("causal") sequence of states.

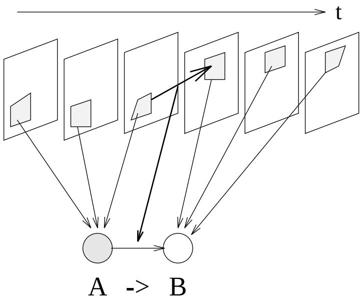

Figure 5: Implication in SG-models. Consecutive measurements of objects are interpreted as belonging to states, generate state labels and formulate state-transitions.

In Fig. 5 the function of such a system is depicted. Firstly, a flow of input data which is supposed to contain measurements of objects is mapped on formal symbols (A and B). Secondly, the timely sequence of these objects is mapped on a formal connection (implication arrow) between the two signs. The names of the states (e.g. A) are, of course, arbitrarily selected by a supervisor.

Thirdly, opponentship is identified with the fact that only one state can be active at a time. This is what we would expect from a typical Newtonian state model, where the states are arbitrarily labeled, see Fig. 6.

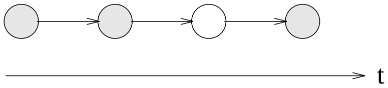

Figure 6: State transition sequence in Newtonian models. At one instant in time only one state is active. Arrows represent state transitions. (States could be labeled $s_1, \ldots$)

3 Dynamic Symbol Grounding: An example

3.1 Architecture

The following neural network example has been implemented to illustrate the theoretical argumentation, it is *not* supposed to serve as yet another SG architecture. (Details of the architecture and the algorithms can be found in [Prem 94b].)

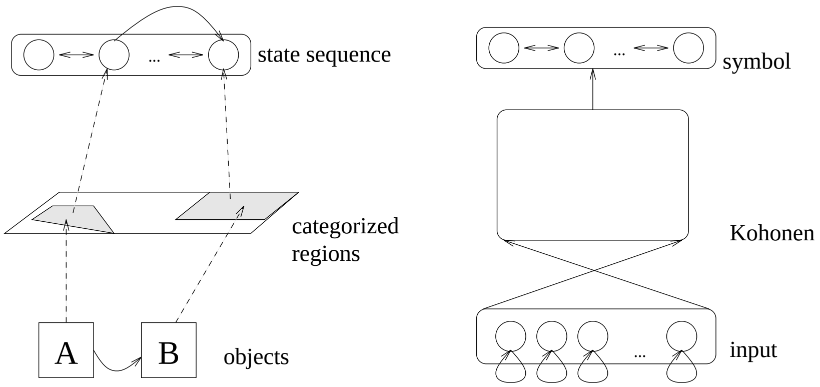

Figure 7: Left: Two consecutive objects are categorized in a Kohonen network and generate a state sequence. Right: The network consists of recurrent input units, a Kohonen layer and a IAC-output layer.

Fig. 7 shows the basic idea and architecture employed in our experiments. Two consecutive objects are categorized by means of a Kohonen network, i.e. their measurements (which can be distorted) map onto different regions of the map. These regions can then be labeled through another layer, the symbol layer. Whereas the formation of categories happens without a teacher, the name for a category (i.e. which unit of the symbol layer should represent the category) is defined by a human supervisor.

The input layer which receives signals from the (simulated) measurements consists of self-recurrent units. The weights of the self-recurrent connections range from 0.0 to 0.7 so as to only slowly change their activation through time and to capture diffent (historical) aspects of the time sequence, cf. [Ulbricht 94].

The Kohonen layer was trained using a variant of the common Kohonen algorithm [Zell 94].

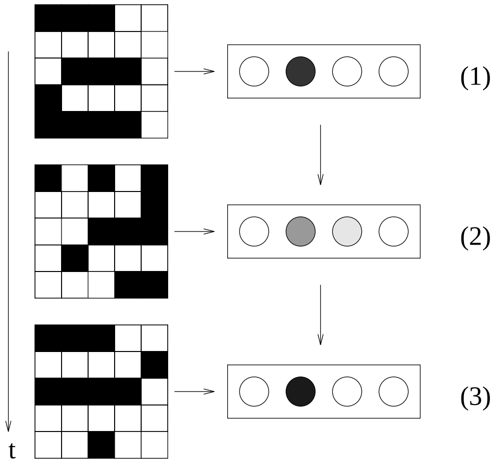

Figure 8: Clear recognition of states in the symbol layer despite of distorted inputs. *Interactive activation* serves to suppress short-term variation of the symbol units, see text.

The symbol layer implements the *principle of the excluded middle* to ensure that only one symbol unit is active per time. This is done by means of *interactive activation* (lateral inhibition [McClelland & Rumelhart 81]) of symbol units. One advantage of this technique is to suppress short-time variations of the symbol due to noise in the input data, see Fig. 8. In addition to the inhibitory connections in the input layer there are also trainable connections which try to capture which active symbol unit follows which.

The symbol layer connections to the Kohonen layer were trained according to a simple Hebbian learning scheme. The intra-layer connections should capture the sequential aspect, i.e. which symbol (state) follows which. In order to capture only really “causal” sequences in the input Hebbian learning between two consecutively active symbol units only occured if (i) the activation of the preceding unit is above a threshold and (ii) if the follwing symbol unit is above a threshold in the very next time step. Condition (i) serves to make sure that only safely recognized states can imply others, condition (ii) ensures that only really implicatons are learned. In order to forget wrongly learned implications all these weights in the symbol layer are reduced by a small

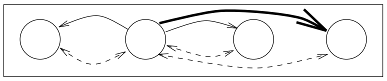

Figure 9: The symbol layer which captures the state transitions. Besides of the IAC connections (dottet lines) other intral-layer connections exist that learn the state graph. The figure shows the connections of the second unit which has obviously often been active immediately before the fourth and so generated a strong connection.

amount in each time step.

It should be mentioned that this architecture is basically a regular SG system (except for the recurrent connections in the input and the intra-layer connections in the symbol layer).

3.2 Experiments

In the experiments with the architecture up to 5 prototypical 5×5 pixel input vectors were used. These pictures were artificially distorted (up to 25%) and presented to the network in a predefined merry-go-round fashion, see Fig. 10. A simple input sequence consisted of showing all 5 noisy prototypes, one after the other, to the network. Each prototype was presented for 3 consecutive time steps ("permanency of objects in time"). It was easy for a network presented with such an input and the sequence of $1, 2, 3, 4, 5$ as the desired output labels to (i) map distorted inputs on symbol units and (ii) to generate the desired state transition graph.

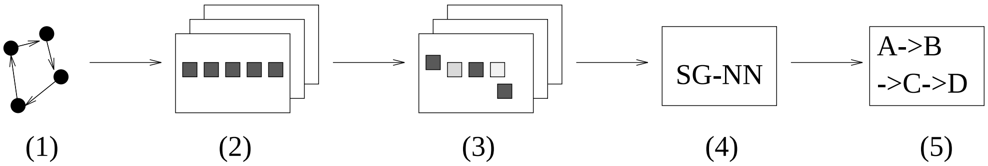

Figure 10: Experimental design. (1) a hypothetical sequence of objects is modeled as a sequence of prototypes (2), which are measured by noisy meters (3). A SG architecture (4) tries to formulate the sequence into a state-transition graph (5).

A more difficult input consists of the following sequence of input patterns: $\alpha \rightarrow \beta \rightarrow \gamma \rightarrow \delta \rightarrow \epsilon \rightarrow \gamma$. The network was trained successfully to label these inputs thruogh the symbol units number $1, 2, 3, 4, 5, 6$. One interesting feature of this architecture is that the recurrent input units allowed the Kohonen net to distinguish between the two same input patterns γ which can only be distinguished through their different context in time. In this case, the state-transition sequence generated in the output layer was as shown in sequence 1.

$$1 \overset{0.95}{\rightarrow} 2 \overset{0.95}{\rightarrow} 3 \overset{0.7}{\rightarrow} 4 \overset{0.95}{\rightarrow} 5 \overset{0.7}{\rightarrow} 6 \overset{0.85}{\rightarrow} 1 \tag{1}$$

It can be seen that the SG system very well models the input sequence, however, has some difficulties in distinguishing between the two same inputs. (Numbers on top of the arrows represent weights between corresponding symbol layer connections.) If the network is trained so as to generate the labels $1, 2, 3, 4, 5, 3$ then object γ is easier to recognize for the symbol layer, but then the state transition sequence becomes something like $1 \rightarrow 2 \rightarrow 3 \rightarrow 1$ and additionally $4 \rightarrow 5 \rightarrow 3$ and another transition $3 \rightarrow 4$. Which obviously reflects that the teacher of the system used too few states for clear and unambigous transitions.

Many variations of this architecture could be studied (but need a more practical context). The aim of the above experiments was only to show how a slightly modified general SG architecure can be used as a automated generator of formal models of informal domains where the state labels are chosen by a supervisor.

4 Teleology and "groundable" Signs

If it is correct that SG is model construction, the necessary next question must be, which sort of models are generated in SG systems. Today's SG architectures can be partitioned into two groups depending on whether they label unsupervised categorizaton or autosupervised backpropagation networks (Fig. 11).

SG has sometimes been suggested to have a system label its internal states (not necessarily found through unsupervised categorization) and to thereby support finding explanations for the system's actions (e.g. [Prem 95b]).

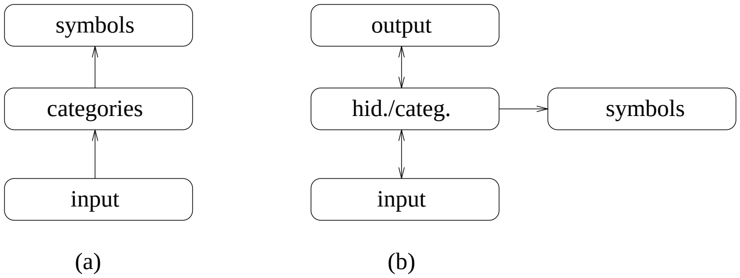

Figure 11: Variants of SG (a) input- and (b) hidden layer categorization.

This would mean to create the system's desired output by means of a supervised learning technique, use representations which generate the desired output for categorization and to then label these categories with arbitrary signs.

It is easy to see that the categories which are generated are now not only dependant on the input, but also on the output. Depending on which task is to be learned by a network with architecture (b) in Fig. 11, other kinds of symbols can be successfully grounded. In another experiment with the modified architecture we trained a backpropagation network to distinguish between the two instances of the same input pattern γ using two output units (01 first occurence of γ, 10 second occurrence, 00 other inputs). The patterns of the hidden layer of this backpropagation network have now been used to generate categories in the Kohonen network and to be labeled by symbol layer units. This time, the architecture could not successfully ground descriptions of the input.

In other terms, in the case of the architecture which categorizes the hidden states of the backpropagation network grounding of the very same set of symbols fails (if the categorization component is forced to find a small set of categories, otherwise some symbols can successfully be grounded). In this case, only labels for the *output* can be grounded, i.e. only labels for the output of the backpropagation network can be correctly produced.

Employing networks which solve another task makes it necessary to switch from input to output descriptions. The symbols which are now grounded do not only describe objects in the measurements, but notions which have to do with the task, which express some kind of "in order to". The reason lies in the fact that the representation which the network has constructed is more a model of the task to solve than of the input. This will alsways happen, when some sort of error minimization training procedure is used. The backpropagation network finds a representation in the hidden layer which is useful for itself, i.e. for its task. Consequently, it now becomes difficult to use these very same encodings of the input as the generator for symbolic descriptions which would make sense to the human observer.

The practical consequence of this result is that if one tries to use the representations generated in the hidden layer of a backpropagation net which is not auto-associative, arbitrary input descriptions (symbols) cannot be grounded. Only those symbols are "groundable" which are in accordance with the goal of the backpropagation network. Without knowing this goal, grounding is hardly possible.

5 Conclusion

The SG architecture presented in this paper can be extended in many ways: One could study differenct ways of processing the input, other categorization methods can be used, and the generation of the state-transition sequence could well be achieved by other means, too. However, such a study does not seem to make sense without a concrete problem, i.e. without a concrete practical domain which is to be modeled automatically. The results presented here do not make claims about the practicality of the presented architecture. They serve to support my proposal that symbol grounding as it is pursued today and as it is argued by Harnad and others is nothing else but the (semi-)automatic construction of a formal model of a non-formal domain.

This need not be a merely negative result. Of course, it suggests that SG cannot solve problems of intentionality (Chinese room) or original meaning. It is also doubtful that it can contribute very much to the semantics of natural language, since it is only based on a very primitive view of correlational word meaning. It can, however, turn out to be a useful and interesting field of research on its own. There are many cases where an automated construction of a world model that remains, at least in parts, understandable for a human, would be desirable. Consider, for example, a robot who is moving in an unknown territory. Having this robot produce a correct set of labels for the environment it experiences or report about an expedition to new territory would certainly be desirable. These are reasons why symbol grounding still deserves further attention.

References

[Chauvin 89] Chauvin Y.: Toward a Connectionist Model of Symbolic Emergence, in Proceedings of the Eleventh Anual Conference of the Cognitive Science Society, Lawrence Erlbaum, Hillsdale, NJ, 580–587, 1989.

[Christiansen & Chater 92] Christiansen M.H., Chater N.: Connectionism Learning and Meaning, Connection Science, 3(4), 227–252, 1992.

[Christiansen & Chater 93] Christiansen M.H., Chater N.: Symbol Grounding– the Emperor's New Theory of Meaning?, Proc. of the 15th Annual Conference of the Cognitive Science Society, Boulder, CO, 1993.

[Cottrell et al. 90] Cottrell G.W., Bartell B., Haupt C.: Grounding Meaning in Perception, in Marburger H. (Hrsg.), GWAI-90, Springer, Berlin, 1990.

[Dorffner 89] Dorffner G.: A Sub-Symbolic Connectionist Model of Basic Language Functions, Indiana University, Computer Science Dept., Dissertation, 1989.

[Dorffner & Prem 93] Dorffner G., Prem E.: Connectionism, Symbol Grounding, and Autonomous Agents, Proceedings of the 15th Annual Meeting of the Cognitive Science Society, Boulder, CO, pp. 144–148, 1993.

[Frixione & Spinelli 92] Frixione M., Spinelli G.: Connectionism and Functionalism: The Importance of Being a Subsymbolist, JETAI Journal of Experimental and Theoretical Artificial Intelligence, 4(1), 1992.

[Harnad 90] Harnad S.: The Symbol Grounding Problem, Physica D, 42, pp. 335–346, 1990.

[Harnad 94] Harnad S.: The Origin of Words, in Durham, W. & Velichkovsky B. (Hrsg.) Naturally Human: Origins and Destiny of Language, Nodus Pub., Muenster, 1994.

[McClelland & Rumelhart 81] McClelland J.L., Rumelhart D.E.: An Interactive Activation Model of Context Effects in Letter Perception: Part 1. An Account of Basic Findings, Psychological Review, Vol.88, 375–407, 1981.

[Nenov & Dyer 94] Nenov V.I., Dyer M.G.: Perceptually Grounded Language Learning: Part 2 – DETE: a Neural/Procedural Model, Connection Science, 6(1), 3–42, 1994.

[Prem 94a] Prem E.: Symbol Grounding Revisited, Oesterreichisches Forschungsinstitut fuer Artificial Intelligence, Wien, TR-94-19, 1994. Presented at the Workshop for Combining Symbolic and Subsymbolic Approaches, 11th European Conference on AI, Amsterdam, 1994.

[Prem 94b] Prem E.: Symbol Grounding: Die Bedeutung der Verankerung von Symbolen in reichhaltiger sensorischer Erfahrung mittel neuronaler Netzwerke, PhD Thesis, Faculty of Computer Science, University of Technology, Wien, 1994.

[Prem 95a] Prem E.: Symbol Grounding and Transcendental Logic, Proc. of the Second Swedish Conference on Connectionism, Skoevde, Sweden, 1995.

[Prem 95b] Prem E.: Understanding Complex Systems–What Can the Speaking Lion Tell Us?, to appear in Steels L. (ed.), The Biology and Technology of Autonomous Agents, Springer, Berlin Heidelberg New York, NATO ASI Series F, 1995.

[Schyns 91] Schyns P.G.: A Modular Neural Network Model of Concept Acquisition, Cognitive Science, 15(4), 1991.

[Searle 80] Searle J.R.: Minds, Brains and Programs, Behavioral and Brain Sciences, 3, 417–457, 1980.

[Ulbricht 94] Ulbricht C.: Multi-recurrent Networks for Traffic Forecasting, Proceedings of the AAAI'94 Conference, Seattle, Washington, 883–888, 1994.

[Zell 94] Zell A.: Simulation Neuronaler Netze, Addison-Wesley, Bonn-Paris-Reading, 1994.