Modelling of syntactical processing in the cortex

Heiner Markert, Andreas Knoblauch, Günther Palm

2007

Preprint:

This is an accepted article published in BioSystems. The final authenticated version is available online at: https://doi.org/[DOI not available]

Modelling of syntactical processing in the cortex

Heiner Markert[1], Andreas Knoblauch[2] and Günther Palm[1]
[1] Department of Neural Information Processing, University of Ulm
Oberer Eselsberg, D-89069 Ulm, Germany
markert@neuro.informatik.uni-ulm.de
palm@neuro.informatik.uni-ulm.de
[2] Honda Research Institute Europe GmbH
Carl-Legian-Str. 30, D-63073 Offenbach/Main, Germany
andreas.knoblauch@honda-ri.de

Abstract. Probably the hardest test for a theory of brain function is the explanation of language processing in the human brain, in particular the interplay of syntax and semantics. Clearly such an explanation can only be very speculative, because there are essentially no animal models and it is hard to study detailed neural processing in humans. The approach presented in this paper uses well established basic neural mechanisms in a plausible global network architecture that is formulated essentially in terms of cortical areas and their intracortical and cortico-cortical interconnections. The neural implementation of this system shows that the comparatively intricate logical task of understanding semantico-syntactical structures can be mastered by a neural network architecture. The system presented also shows additional context awareness, in particular the model is able to correct ambiguous input to a certain degree, e.g. the input "bot show/lift green wall" with an artificial ambiguity between "show" and "lift" is correctly interpreted as "bot show green wall" since a wall is not liftable. Furthermore, the system is able to learn new object words during runtime.

Keywords. Spiking associative memory, distributed representation, sparse patterns, neural assemblies, language understanding, global brain modelling

1 Introduction

This paper describes work in progress on the integration of vision, language and action in a cortex model and its demonstration on a small robot (ActivMedia PeopleBot). It has been part of a cooperative European project (MirrorBot), but in this paper we focus on the work carried out at the University of Ulm, and in particular on the aspect of language understanding.

We begin by describing the scope of the whole project that will be pursued in our group in Ulm for a few more years.

The idea for this project is based on three previous lines of research:

1. Hebb's old idea of *cell assemblies* formed by Hebbian learning and providing the basic representation of objects and thoughts in the brain [1] has been worked out in more detail at the Max-Planck-Institute in Tübingen [2][3][4][5][6][7] and has led to mathematical investigations on the information storage capacity of associative memories showing their practical usefulness and the importance of sparseness of the activity patterns to be stored [3][8][9]. This research encouraged us to pursue the idea that the cortex might be a huge associative memory (in addition to some other functions of specific cortical areas), or more exactly that it is composed of auto-associative memory modules representing different aspects of objects in different areas and hetero-associative long-range connections between these modules [10]. This leads directly to the assumption that at least a reasonable proportion of all synapses in the cortex are capable of Hebbian synaptic plasticity [11]. Of course, these ideas are rather old, but only now[1] it is possible to simulate and test these ideas on a large-scale involving several cortical areas and to gain some insights into the human brain areas involved in language understanding and production and visually guided action.

2. One of the basic problems with Hebb's idea of cell assemblies is the *superposition* or *binding problem* [12][13][14][15]: If a superposition of several assemblies is activated in the cortex or in some cortical area, how can it be disentangled into its components? One answer to this problem could be to avoid an activation of superpositions in the first place, so that normally only one assembly is activated at a time. This, however, would be a serious restriction to the practical usefulness of assemblies in handling more complex computational problems like grammatical correctness. Clearly this has to do with the issue of compositionality (for example of language) [16]. To address these issues, it seems necessary (but may be not sufficient) to create a cortical architecture that can solve the superposition problem. It turns out that a simple bidirectional connection of two cortical areas can solve this problem in a very natural way [17][18]. After we found this simple solution, we were encouraged to tackle larger problems involving the compositionality of language or action planning.

3. Another idea closely related to Hebbian cell assemblies is Abeles' idea of *synfire chains* [19][20]. Clearly, a synfire chain can be considered as a sequence of heteroassociations in a feedback network, whereas an assembly is simply an autoassociative feedback on itself [4]. So Abeles' work could be interpreted as an indication that in some areas of the brain the local feedback connectivity between excit-

1 this is defined with an uncertainty of at least five years

atory neurons is used not only for auto-associative stabilization of assemblies, but also for heteroassociative formation of synfire chains. The experimental evidence seems to support the fact that this is more prominent in the frontal part of the cortex (see the discussion of "patterns" as evidence for synfire chains, e.g. [21]). Since heteroassociative synfire-chains are ideally suited for storing and recognising sequences, we were encouraged to try this mechanism for language understanding and production and to find out how far we can get with it. In this enterprise we also cooperated with Friedemann Pulvermüller, who had similar ideas [22][23].

Based on this work and encouraged by recent experimental evidence [24][25][26], we created a cell assembly based architecture that combines the simulation of a large number of "modules" or "areas", each containing a large number of spiking neurons and each representing a different aspect of an object (such as visual, auditory, motor, syntactic or semantic aspects). The connections in this network are formed by association or by simple Hebbian learning. The details of the neuron model and the learning rule are given in the appendix. The complete layout of all modules comprising our system (as of today) is described in [27]. In order to demonstrate the functionality of the cortical network we have embedded it into a simple robot scenario. To this end, we have to create visual and auditory input representations and motor output representations. We use a video camera and a hierarchical neural network classifier [27] to create a neural representation that is activated in the primary visual area of the model. Similarly, we plan to use a microphone and speech recognition to create a neural representation to be activated in the primary auditory area. Currently, however, we simply create this representation from typed language commands. We have created random sparse motor-representations for the commands in the "motor"

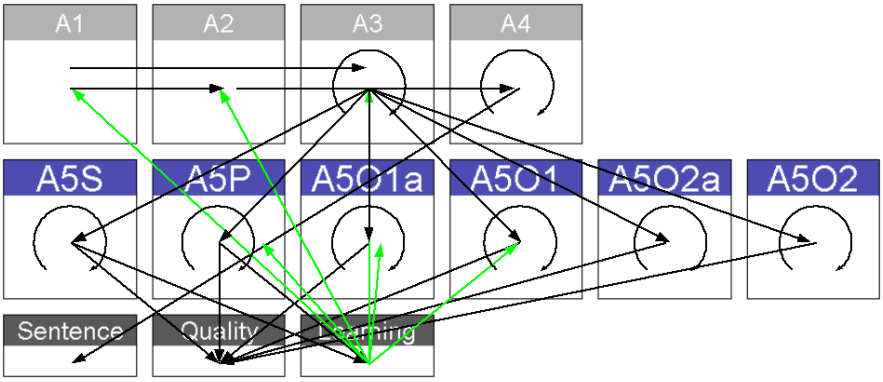

Figure 1: Architecture of the cortical language model. The language system consists of 10 cortical modules (large boxes), two control modules ("Quality", "Sentence" areas at the bottom) and several thalamic activation fields, where only one of them ("Learning") is shown. The straight arrows correspond to heteroassociative intra-areal connections, circular arrows correspond to short term memory.

area of the model; these are translated directly into motor commands for the robot. Presently the cortical network simulation runs on an additional PC, while the visual object recognition, the handling of input and output commands, and the control of the robot is performed by the PeopleBot's on-board PC.

In the following we describe in detail the language system of our architecture (Figure 1). The model consists of several language areas which make it possible to parse and grammatically analyse spoken or typed language input. Basically, different cortical modules (areas) in the model represent different aspects of the same entity, e.g. auditory language, semantical or syntactical aspects. Cortical areas are interconnected with mostly bidirectional corticocortical long range projections realised by heteroassociative memories that translate between the different aspects or the corresponding representations.

The language system presented here is capable of parsing a stream of single words, analysing it with respect to a certain regular grammar, and to "understanding" a sentence by translating the word sequence into an action representation constituted by a global cell assembly extending across several visual, pre-motor and motor areas [28]. Furthermore, our language model is capable of representing and resolving ambiguity from a broader context, i.e. if it is not possible to interpret a word in a unique way, several alternatives can be kept active until additional context information can be used to resolve the ambiguity. For example, the sentence "bot lift bwall" where the word "bwall" is a mixture between ball and wall that was not correctly understood, the system can resolve this ambiguity to "bot lift ball" because a wall is not liftable. In a broader context, it could as well be another information from the whole sensory-motor context that provides the disambiguating input.

The system can incrementally learn new patterns during performance. When the associative memories get a "learn" signal, they enable online learning and can create new representations if the input does not match any of the previously stored patterns. The heteroassociative connections are then trained accordingly. Each area and heteroassociation in the system is capable of learning new patterns online, but currently we implemented only the learning of novel object words.

2 Neural Associative Memory

Each area in the network depicted in Figure 1 is modelled with a variant of the so called spike counter model of associative memory [17][29]. The spike counter model is based on Willshaw's binary associative memory [30], but extends it with a more sophisticated retrieval algorithm that allows for much better pattern separation if the memory is addressed with a superposition of patterns. We have chosen the Willshaw model as basic system because it is a biologically plausible while still simple implementation of the idea of cell assemblies. It is also very efficient in terms of storage capacity and information efficiency [8][31][32][33]. Our model uses a spiking version of the Willshaw model [17][29] which adds the possibility to measure spike time coincidence.

Here, we use a rather technical implementation of this model which still allows for fine measurement of spike timing and especially of the temporal order of the spikes. The neurons are of a simple integrate-and-fire type [34][35] with reset. To further simplify calculations, we introduced global time steps, roughly corresponding to one time step within the binary Willshaw model [30], and finer relative time steps within each global step that allows for an exact representation of the spike times. In one global time step, every area calculates one pattern retrieval with respect to the relative time scale. Every neuron is allowed to spike at most twice per global step and the step ends as soon as no more neuron is able to spike anymore. The algorithm necessarily terminates because at a given time, either all neurons already spiked twice or some spiked twice and others have constantly decreasing membrane potential while not receiving any more input which ensures that these neurons will not spike dur-

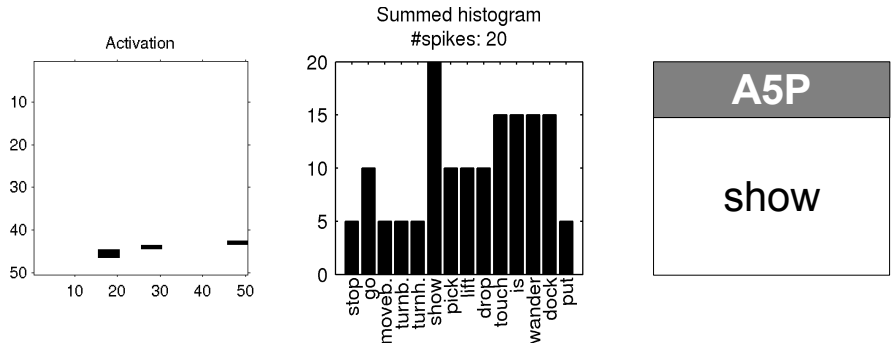

Figure 2: From neural activation to a pattern display: Each cortical module is an associative memory which is trained with a set of named patterns. On the left hand side, the neural activation is displayed. The module consists of 2500 neurons that are arranged 2-dimensionally as a square of 50 times 50 neurons, the axes give the index of the neurons, black dots mean activation. If a set of neurons is active after one global time step, a histogram over the stored patterns is calculated (middle). The pattern names which match best are then displayed as a short and intuitive description of the activation in the cortical module (right).

ing the whole global step. When one global step ends, the resulting spike trains are propagated through the heteroassociative connections and the input for the next global step is prepared. Then, the neurons' potentials are reset and the next global time step starts. A more detailed description of the model is given in the Appendix and in [36]. See also [27][29] for more information on the Spike Counter model approach.

This simplified model combined with binary Hebbian learning, sparse patterns and sparse activation in the whole system allows for efficient computation [9] making the system fast enough to control a robot in real time.

For demonstration purposes we use a special way of displaying neural activity in the system (see Figure 2). Instead of showing neural activation directly, we display the names of patterns that match best. When the system is started, the associative memories are trained with a fixed set of name/activation pairs, i.e. each assembly (i.e. activation pattern) has a corresponding text description. After each global time step, the result of the pattern retrieval is displayed as shown in Figure 2: in the first step, a histogram is calculated which measures for each stored pattern how many neurons belonging to it are active. In the next step, basically all pattern names that have a large number of active neurons are displayed. There are additional constraints, e.g. we display only those pattern names that have at least 80% of their neurons active and of course it is possible to display the completeness (i.e. the percentage of active neurons) in the output (not shown in the figure). It is also possible to switch the display between the different modes while the system is running. In this way, one can get a quick overview of the state of the network even with many simulated areas (our large model with language and action planing has about 50 areas).

3 Cortical Language Model

The language model consists of a standard Hidden Markov Model (HMM) [37][38] based speech recognition system and a cortical language processing system which is capable of interpreting streams of words with respect to a regular grammar [28][39]. Currently, the HMM speech recognition is not yet integrated into the system and therefore, input must be typed on a keyboard.

Figure 1 gives an overview of the cortical language processing system. Each area is modelled as a spiking associative memory [17] of 400 to 2500 neurons (the neuron number per area depends on the number of stored cell assemblies and the required fault tolerance). The areas store binary patterns in local synaptic connections by Hebbian learning, forming a neural assembly for each stored pattern. The areas are connected via heteroassociative memories depicted as arrows in Figure 1.

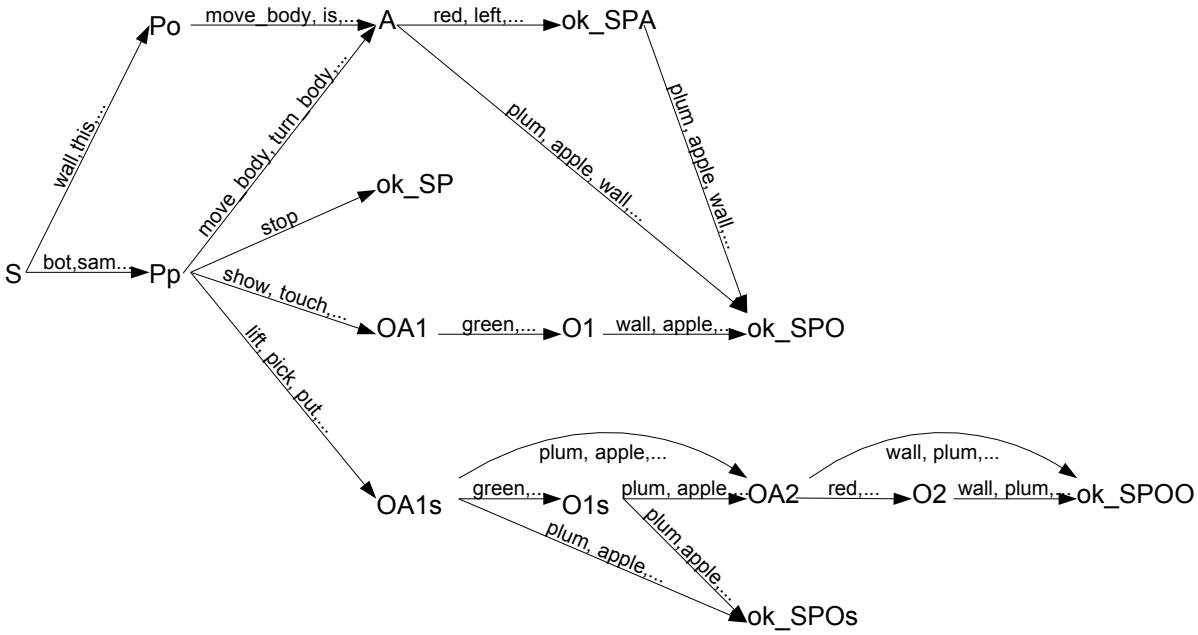

Figure 3: Contents of the sequence memory A4. The symbols in larger font (S, Pp, Po etc.) represent the random patterns stored in area A4 and correspond to states of a finite state machine. The arrows are heteroassociations from A4 onto itself and represent the possible state transitions. The words printed along the line are example inputs that can lead to the state transition denoted by the arrow. Not shown in the figure are additional error states that can be reached from every state within the system if a non-matching input occurs.

The model roughly consists of three parts:

Auditory areas: Areas A1, A2 and A3 are the auditory areas. A1 processes the output from the speech recogniser or alternatively typed commands (e.g., phoneme based word representations). A2 and A3 represent syntactical and semantical aspects of the word, respectively (e.g., the overlaps of the A3 representations reflect semantical similarity relations of the words).

Grammatical areas: Modules A4, A5S, A5P, A5O1a, A5O1, A5O2a and A5O2 mainly serve grammatical functions. A4 works as a sequence memory and basically represents the regular grammar the system can understand [40]. Modules A5X store words with respect to their grammatical role, i.e. they classify into subject, predicate, attributes and objects.

The system is able to understand regular grammars. In the current implementation, the cortical language processing system is able to parse two main sentence types, namely “subject predicate object” (SPO) and “subject predicate adjective” (SPA). There exist several subclasses of this like SPOO or SP. See Figure 3 for an overview

of the sequences stored in area A4 that represent the grammar the system is able to parse. The following example sentences are all valid with respect to our grammar and can be correctly parsed by the language system:

“bot show plum” (standard SPO), “bot show green apple” (SPO with attribute), “where is plum?” (SPO), “this is plum” (SPO), “this is green” (SPA), “wall is red” (SPA), “bot put apple plum” (SPOO), “bot put red plum yellow lemon” (SPOO with attributes).

The system is also able to resolve ambiguous inputs, e.g. the sentence “bot lift green bwall” with an ambiguity between “ball” and “wall” in entered, the model is able to correctly resolve this to “bot lift green ball”, as a wall is not liftable. Also, “bot show/lift red ball” (with an artificial ambiguity between lift and show) is correctly interpreted as “bot lift red ball”. However, the sentence “bot show/lift green bwall” is not perfectly dealt with. The system tries to disambiguate the sentence and depending on the evidence for each of the alternatives, it might end in a state like “bot show/lift green ball” where it lost the possibility for the object “wall”. Another example the language system is not yet capable of understanding is “bot show orange orange” where the two words “orange” have identical representation in the auditory areas. We expect that it is possible to understand such sentences with the mechanisms used in our architecture, but it might require changes to the grammar and additional use of the disambiguation mechanisms, because the “bot show orange orange” example requires to resolve ambiguities on the grammar level. The first part “bot show orange” might be interpreted as SPO as well as SPA sentence, in both cases the sentence can already be complete, requiring additional disambiguating information (which could for example also be the fact that there is no orange (fruit) in the visual field). If the additional word “orange” comes in, it becomes clear that it has been a SPO-sentence with an additional attribute, but to understand it, the intermediate ambiguous state needs to be represented, which the model is currently not able to do (because it cannot yet handle disambiguities in the A4 module).

The examples above show that our grammar is simplified compared to natural language as it does not contain any prepositions. Although the number of different sentence types is not very large in our example implementation, it could be quite easily increased (basically, the sequence memory A4 needs to store more sequences). However, we did not yet implement a more complicated grammar in our current system.

The grammar shown in Figure 3 is able to distinguish between small and large objects. This will result in an error state if a sentence like “bot lift wall” is entered into the language processing system. To achieve this, the grammar distinguishes different SPO-like sentences (one type for large, one for small sentences, calles “SPO” and “SPOs” respectively) and different action words let the system decide between the different paths (e.g. “bot

lift" will drive the system into the "SPOs"-path, as the robot can only lift small objects, where "bot show" will choose the "SPO"-path because the robot can show any object, see also Figure 3). There are no additional semantical checks implemented in the grammar (but some disambiguation information which is described below).

Additional primitive areas: The other areas in the system represent internal states or serve as thalamic activation fields. The Sentence/Quality areas represent internal states, where the first becomes active if a complete sentence has been parsed and the second represents the quality of the auditory input, which becomes bad if it was not possible to unambiguously interpret the input. The "Learning" area basically is an activation field which activates special *learn signals* in certain areas if learning of a new word is required (indicated by a "this is" sentence). There are additional primitive activation fields that are not shown in Figure 1. They basically inhibit special areas, for example each of the A5X-areas has a corresponding underlying activation field that controls its activation.

When a sentence is processed, the auditory input activates a corresponding representation based on linguistic features such as phonemes in A1. Then, the word becomes classified with respect to function in A2 and content in A3. Area A4 is used to emulate a deterministic finite automaton (DFA) which represents the regular grammar the system is able to understand [40]. Its state gives information about the type of word that is expected as the current input in order to form a grammatically correct sentence. The system then activates one of the areas A5X which matches the internal grammatical state represented in area A4, i.e. if the word is supposed to be a subject, A5S becomes active. If the currently processed word in area A3 matches the word type required in the active A5X area (i.e. the heteroassociation between A3 and A5X finds a matching pattern), the corresponding representation in the area A5X becomes active. Otherwise, the input was not grammatically correct and the automaton realised in A4 switches to an error state.

4 Disambiguation

The system is able to detect and correct ambiguous input on the single word level as long as enough context information is or becomes available. In order to resolve ambiguity when enough context information becomes available, as a first step it is necessary to represent the ambiguous state for a certain time. This is done by adjusting the pattern separation strength of the concerned area. With high separation strength, the retrieval inhibits neurons that do not belong to the same pattern as the ones that are already active and thus only one pattern usually gets activated. With lower separation strength, the inhibition is weaker and it becomes possible to activate several patterns in the same retrieval step. Thus, lower separation strength allows for representing ambiguity by

simply activating a superposition of several possibilities. Spike time information is helpful at this stage, because the stronger activated patterns fire earlier, which helps to resolve the ambiguity later.

During retrieval, each area calculates a quality measure (see Appendix for details) which becomes high if the address pattern matches one of the stored patterns very closely and it becomes low if a unique decision for one output pattern is not possible. A simple feedback loop from the quality measure to the separation strength parameter then enables the area to automatically control the handling of ambiguity: If the input matches a stored pattern closely, separation strength becomes high and the addressed pattern becomes activated; if the input is a superposition of several stored patterns, separation strength becomes lower and several patterns can be activated in the same retrieval.

5 Incremental Learning of New Objects

Currently, the network architecture (including stored patterns, connections and connection weights) of the language processing system is initialised from a parameter file when the simulation starts and (with the exception of object word representations and heteroassociative connections between several areas; see below), all parts of it stay fixed throughout the rest of the simulation. However, our system is capable of learning new objects while performing. For the language model described here, this means that new representations for the object words need to be generated. Details on the generation of new representations are given in the Appendix.

Currently, learning is triggered by a special command ("this is X", where X is a novel object word). If the system encounters a previously unknown object in this situation, a new sensory representation will be learnt in module A1. Then, this new representation acts as input to the subsequent modules in the language part and triggers the generation of new representations and heteroassociative connections there (see Appendix). In the complete system where the language model is also connected to an action planning and object recognition part, learning of a new object also means to learn a new visual representation of the object in the pattern recognition network and the visual input area and then to learn the visual processing path until a comparison module is reached where a common representation for the visual and auditory aspects of the objects is generated [27].

Learning is initiated by the special command "this is X" where "X" can be a word that is new to the language system (i.e. that has no representation in A1). The language system realises the command "this is" and activates a special learn signal (area Learning in Figure 1 becomes active). With active learn signal, a bad retrieval quality in the language areas will not be interpreted as uncertainty in the auditory input but as evidence that a previously

unknown object is going to be learnt. If the latter is the case, new representations for the object will be generated in the corresponding area. If required, heteroassociative connections to and from this area are updated accordingly (details are given in the Appendix). A new assembly is always generated from the most strongly activated neurons in the area where learning takes place, while some additional randomly chosen neurons may also be added to allow for greater variance of the patterns.

After successful learning, the new object can be used and understood like any of the previously stored objects. Thus the system can correctly understand a sentence like “bot show cup” after “cup” has been learned, i.e. after the sentence “this is cup” accompanied with pointing to or presentation of a cup has been processed.

6 Results

We have implemented this language system on a PeopleBot robot. Running on a standard laptop machine (Pentium M 1.5 GHz), it is able to process sentences faster than one can actually speak or type the commands so that the system meets real time constraints. Presently, the language system consists of 18000 neurons in total and has a vocabulary of about 50 words. Our current implementation does not yet have a speech recogniser, so input must be typed in with a keyboard. Alternatively, it is also possible to directly activate several neurons in the input populations.

In the following we will show in detail how the system deals with the ambiguous sentence “bot show/lift green wall” with an artificial ambiguity between “show” and “lift” in the verb input (parts of both representations get activated in the input of area A1 with a small bias towards “lift”). Figure 4 shows the system after the word “bot” has been processed. Area A1 is the input area where the assembly representing the word “bot” is active. For convenience, the name of the active representation is displayed instead of neural activity. The activation has been forwarded to areas A2 and A3 which separate between simple grammatical structure elements (syntax) and the meaning of the word (semantics). Area A4 has the S-assembly activated, which is the start symbol for the grammar (see also Figure 3). In our special grammar, all sentences start with a subject, so the symbol “S” might also be interpreted as the expectation of a subject. This in turn activated area A5S which now represents the subject of the sentence.

Figure 5 shows the system after the second word, “show/lift”, has been processed. We activated a mixture between show and lift in area A1. A2 realised that it does not know this word, lowered its separation strength accordingly and finally activated a superposition of lift, pick, drop and show. The words pick and drop show up be-

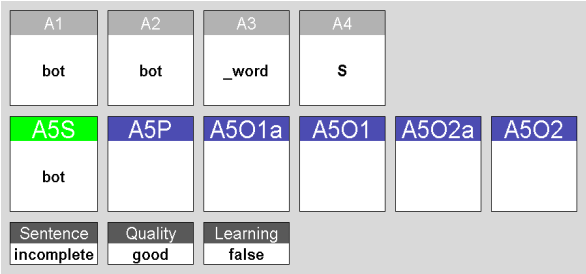

Figure 4: The language system after processing the input word "bot" as start of a new sentence.

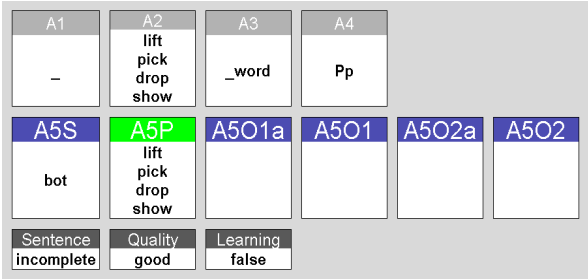

Figure 5: The language system after processing "bot show/lift", where "show/lift" is an artificial ambiguity between the words show and lift.

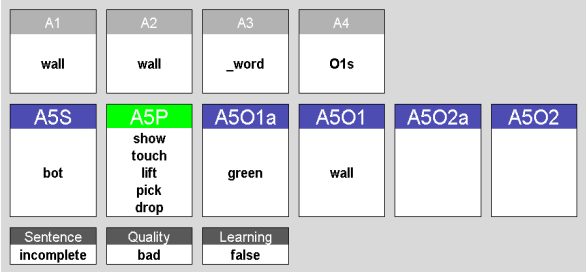

Figure 6: The language system while disambiguating "show/lift". The input "bot show/lift green wall" is completely processed in A5X areas and the disambiguating connection from A5O1 to A5P is starting to resolve the uncertainty.

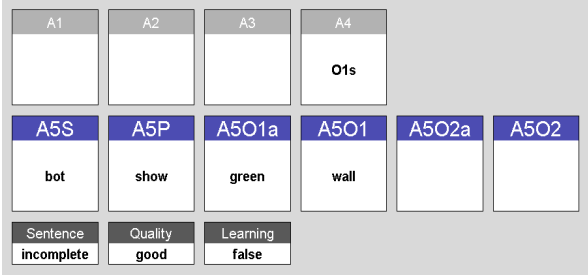

Figure 7: The language system after successfully disambiguating "bot show/lift green wall". Note that the Quality field shows "good" again, indicating that everything was understood in a unique way.

cause they have a larger overlap with lift than show has. As the associative memory is completing patterns independent of the separation strength, the pick and drop patterns are completed as soon as the separation strength gets low enough to allow for it (which it must to allow for activating the show assembly). As before, the information is then processed until area A5P represents the ambiguous verb of the sentence.

Some global steps later, the sentence “bot show/lift green wall” completely arrived in the A5X areas as shown in Figure 6. As soon as the “wall” representation becomes active in A5O1, it starts resolving the ambiguity via a weak connection from A5O1 to A5P. This connection now supports all action verbs that can act on a large object like a wall, resulting in the touch assembly to also show up in A5P.

In the following few steps the disambiguating input from A5O1 to A5P continues to support verbs that can be applied to large objects, in particular supporting the “lift” assembly in A5P. This in turn increases the quality

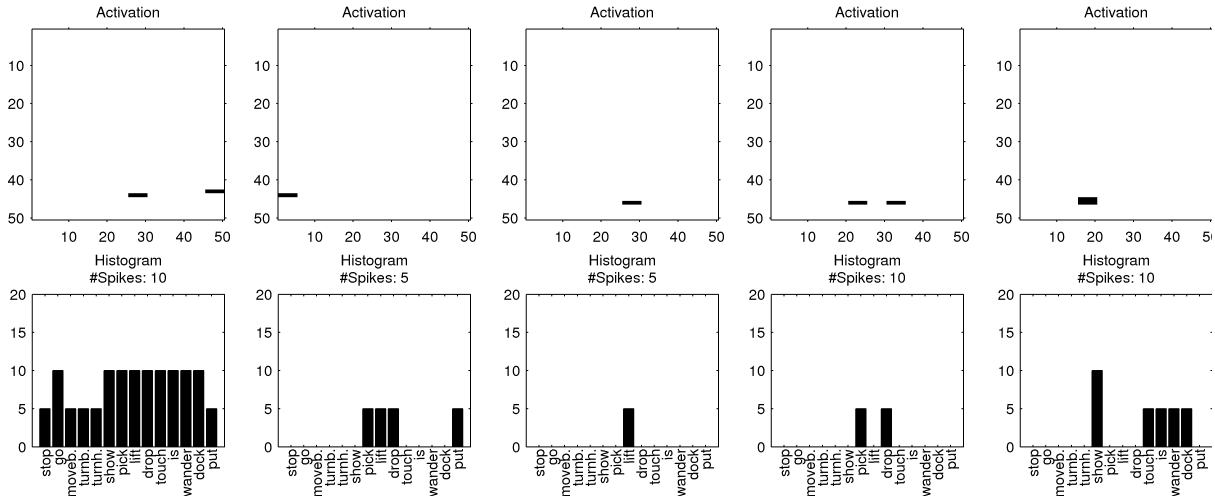

Figure 8: Details of global time step 26: In the first row neural activation is shown, in the second row the corresponding histograms over the stored patterns are given. The figures have to be interpreted as described at Figure 2. Relative times are around 407, 517, 690, 739 and 913 in the columns from left to right respectively. Each pattern consists of 20 neurons, where the patterns have large overlaps. The total number of spikes for each column is displayed above the histograms. Although only ten neurons spiked around relative time 407, there are many patterns with ten active entries in the corresponding histogram because the neurons characterise some feature that is present in many representations. Typically, overlaps spike first as they get the most input. Note that neurons can spike at most twice in the whole global step. We do only count the first spike of each neuron for the figure.

measure, and area A5P increases its separation strength gradually. Finally, this allows for completely resolving the ambiguity as shown in Figure 7.

We will now describe in detail how the disambiguation takes place. Figure 8 gives an overview of what is happening in area A5P in global step 26, which is exactly one global step before the situation depicted in Figure 6 (for A5P, this is the same situation as shown in Figure 5). The figure gives an overview of the neuron activity at the relative time steps in the retrieval where some neurons are spiking. Note that the absolute times have no clear relation to with any real world time scale and therefore are given without units here. In the situation shown, A5P is representing the ambiguity between show and lift which also activates the assemblies for pick and drop due to the pattern overlaps. The first column shows the status of A5P around relative time 407. Ten neurons spiked and from the histogram over the patterns, one can see that many of the verb patterns actually share these neurons. Basically, five of the neurons code for the property “verb”, another five encode the property, that the correspond-

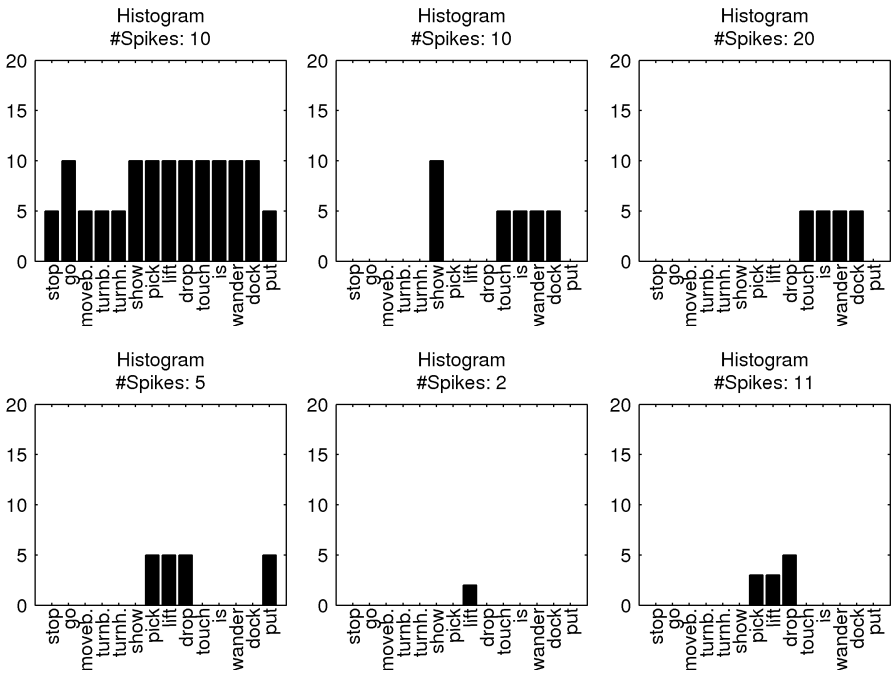

Figure 9: The spike histograms of global time step 27. The same displaying technique as in Figure 8 is used. The relative times are 195, 265 and 389 for the first row, 518, 739 and 913 for the second row. As no neuron can spike more than once, the neurons leading to the histograms are all different and thus the histograms can simply be added to get an overview of how many neurons are active for one special pattern.

ing actions can be done with exactly one object. The next neurons that spike around relative time 517 are depicted in column two. They basically encode that the action can only be done with small objects and not with large ones like e.g. a wall. The next set of spikes in column three (relative time 690) completes the lift-assembly, the five neurons are unique for this pattern. At relative time 739 shown in the forth column in Figure 8, the pick and drop assembly get completed due to the autoassociative feedback connection. Five of the neurons firing at that time belong to the pick assembly, the other five belong to drop, making both of the patterns unique. The last ten spikes in global step 26 around relative time 913 finally complete the show assembly. Five of them are unique to show, another five encode the property that the corresponding action can be done with large objects.

The address pattern that led to the ambiguous representation of show and lift in A5P has a slight bias on lift, which makes the lift assembly spike first. The overlap of patterns typically gets the most input because it has heteroassociative input connections from many patterns in the addressing area. In the first two columns in Figure 8 only neurons representing overlaps become active, where the neurons in the second column are less common ones which do not have as many input connections and thus spike a little bit later. In the third state depicted, the

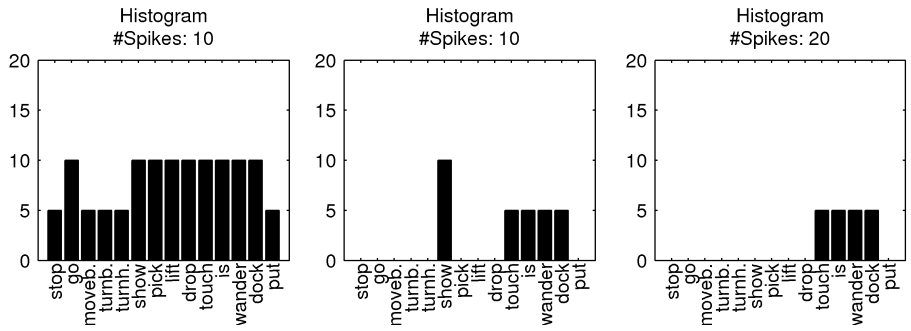

Figure 10: The spike activation in global step 28. The same displaying technique as in Figure 8 is used. Relative times of the columns from left to right are 128, 155 and 196, respectively.

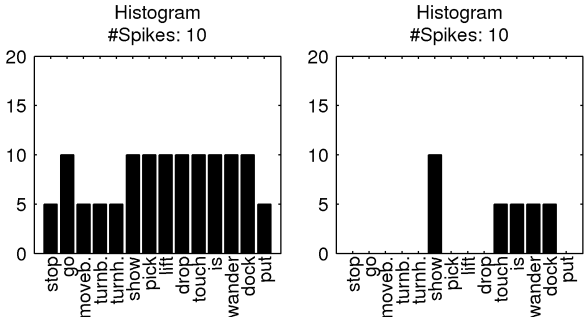

Figure 11: Global time step 30. The same displaying technique as in Figure 8 is used. Disambiguation is finished, only the show assembly is fully active. Relative times are 76 for the left hand side and 85 for the right hand side histogram.

lift assembly is completed according to its higher activation in the beginning. The next set of spikes are triggered by autoassociative connections completing the patterns “pick” and “drop” because of the low separation strength at this stage. The last spikes finally complete the show assembly.

Note that setting the separation strength high enough (i.e. increasing inhibition in the area) to avoid the completion of pick and drop would also forbid the neurons spiking in the last column of Figure 8 to become active and thus, the show assembly would not get completed.

Figure 9 gives an overview of the spike activity in global time step 27 which is depicted in Figure 6 also. The disambiguation is starting to work and in the first step, all actions that can be done with large objects like a wall get additional input. This makes the show assembly the first to be completed, several others follow. Especially, the touch assembly also gets completely activated due to the additional input and the very low separation strength in this step.

Global time step 28 is shown in Figure 10. The disambiguation is starting to resolve the problem, the lift assembly does not show up any more, the show assembly is the first that is completed. The actions possible with large objects gain some more strength through the disambiguating input from area A5O1 ("wall") and still get completed in the last set of spikes. In global step 29, the situation does only change quantitatively, only the relative times change a bit compared with step 28 due to the disambiguating input. The order of the pattern completions does not change compared to step 28. The separation strength is gradually increased.

Disambiguation almost finishes in global step 30 depicted in Figure 11. The first spikes belong to the overlap with other patterns, the following spikes complete the show assembly. No other pattern gets completed in this step and separation strength returned almost to its maximum value. In the following global step 31 (shown in Figure 7), disambiguation is completely finished and separation strength returns to maximum value. The spike activity looks very similar to that in step 30.

7 Discussion

We have realised a large-scale cortical model which uses the interaction of several cortical areas to achieve a syntactical analysis and semantical understanding of simple sentences (mostly commands) and to organise the appropriate responses to these sentences.

The neural implementation of this language understanding system not only shows that this comparatively intricate logical task can be mastered surprisingly well (see [41][42]) by a neural network architecture in real time, it also gives some additional advantages in terms of context awareness and robustness. The system can correct ambiguous input on the single word level due to context information of the whole sentence and in a larger model even with respect to the whole sensory-motor situation. Similarly, the language input could be used to disambiguate ambiguous situations in other fields, e.g. in visual object recognition. We have illustrated the process of disambiguation in detail for a specific example sentence.

Our approach tackles a long-standing problem in artificial intelligence and brain theory, namely the understanding of language. Therefore it is clearly related to a lot of scientific work, particularly in the areas of neural modelling, artificial intelligence, and linguistics.

With respect to neural modelling, our work differs from most other work by the global functional approach to brain modelling (see [43][44] for a related approach, also related is the NOMAD project, see e.g. [45]). Our cortical language system can better be understood in the context of a larger model that covers many cortical areas

and integrates language understanding, visual object recognition, visual attention, and action planning. Most neural models deal only with one or two of these aspects at a time (only vision, hearing, action, including more specific details of one or two cortical areas), the visual system being modelled most intensively (e.g. [46][47][48][49]). More complex models that deal for example with working memory [50] or with the replication of fMRI activations of multiple areas in complex tasks [51] deal with the brain at a purely boxological and essentially non-functional level, i.e. without any detailed understanding of how the task is actually achieved (computationally) by the neural network.

Related work in classical artificial intelligence dates back to the 70's [52] and suffers from the lack of a connection to the neural processing level. These approaches are well suited for abstract planning (e.g., the “tower of Hanoi”), but encounter serious problems when they have to work in real-world applications due to problems of uncertainty and inaccuracy which can be handled in our system by means of distributed representations, which make it possible to use some notion of similarity. These problems are related to the so called problem of "symbol grounding" and are tackled in a number of "connectionist" or "hybrid" approaches (e.g. [53][54][55][56][57][58][59]) that are similar in spirit to our approach but less directly related to biologically realistic spiking neural models and anatomically plausible neural network architectures. Some of these approaches have also been used for language understanding [60][61][62], but lack the embedding into a visuo-motor system. Also in many cases it would be hard if not impossible to extend the approach from the toy problems that are presented to more realistic situations.

In computational linguistics many papers rely on the artificial intelligence approaches mentioned above and try to produce a kind of parser that can distinguish grammatically correct from incorrect sentences for much more sophisticated grammars (e.g. [41][62][63]), but without considering the closely related and from our point of view inherently intermingled semantic aspects and without trying to demonstrate an "understanding" of sentences. Of course, we can only do this by first restricting ourselves to very simple command sentences and demonstrate understanding by performing the commands. This idea is clearly related to the recent philosophy of embodied computation [64][65][66], which emphasized the necessity to embed the computations in autonomous real world robotic agents. Also, we have already shown some examples which go beyond this simple paradigm (learning of new objects). Our grammar can be easily extended by simply storing more sequences in the sequence area A4 (Figure 3), our vocabulary can be extended by storing more words and their different sensory representations in areas A1, A3 and A5X. The storage capacity is in principle only limited by the number of Hebb synapses in all these networks (it is proportional to it with a reasonably large factor; see [3]). We have

shown that a sequence area can in principle be viewed as a distributed neural realisation of a finite automaton, which in turn can be made to recognize any regular grammar [67]. With a little more imagination one can even realize a neural push-down automaton [22] which would be a more natural way of recognizing embedded structures like relative sentences (i.e. context-free grammars). However, even such a neural push-down automaton would have a limited depth such that its computational capacity remains the same as that of a finite automaton. Essentially, this also seems to be the limit of the practical human language capacities (without extensive use of paper and pencil [67]).

The idea of using artificial neural networks to represent and train finite automata and even some extensions of these, is of course not new and has been discussed to some extent in the connectionist and artificial neural network literature (e.g. [68][69][70][71][72][73]). But here the problems were usually more of the symbolic type.

Although we can use the associative sequence memory A4 to store more grammatical sequences and although we can demonstrate the learning of new word-meanings, we clearly do not address the problem of language acquisition, i.e. the circumstances of social and brain organisation that are required to set up a learning procedure that can result in the creation and sequencing of new "grammatical representations" in an area like our A4. We know of only very few projects that are close to our own approach in most of the aspects discussed so far ([74][75][76][43]). The most obvious next steps we want to take in this project are to incorporate speech recognition, to extend the vocabulary and the grammar, and to generate output sentences.

8 Appendix (Mathematical Description of the Model)

8.1 Neuron Equations

In the following, we give a short introduction of the model used. For a more detailed description, see [36].

The cortical language model consists of several cortical modules, each of which has the same underlying mechanisms. Thus, where possible, we describe the model only for one cortical module. As mentioned earlier, the model exhibits two different time scales, namely global time steps and relative time steps subdividing each global step. We call the global steps s and the relative time t. For simplicity, we consider a fixed global step s here. The same mechanisms apply for all other global steps.

A cortical module consists of N neurons, where the number of neurons might be different for different modules. The state of a neuron is basically given by its membrane potential x (details will be given below), where a spike

is emitted whenever the membrane potential exceeds a global threshold Θ. The output vector of the cortical module at global step s is defined as $y_i^s(t)=\mathbf{1}_{x_i(t)\geq\Theta}(t)$, where i is the index of the neuron. Each neuron is allowed to spike at most twice within one global time step. After the first spike the membrane potential is reset to zero. We define the instantaneous spike rate vector of the cortical module as $r_i(s,t_{max})=min\{0<t\leq t_{max}: y_i^s(t)=1\}^{-1}$, where the rate is defined to be zero when the minimum is empty or $t_{max}\leq 0$. We will sometimes refer to the case $t_{max}=\infty$ which means the inverse of the first spike time of a neuron.

Within each cortical module, there is an autoassociative coupling which is given by the matrix

$$A=\begin{pmatrix} a_{11} & \dots & a_{1N} \\ \vdots & \ddots & \vdots \\ a_{N1} & \dots & a_{NN} \end{pmatrix}$$

where the entries a_{ij} are binary numbers, i.e. zero or one. The matrix A is built exactly like the autoassociative matrix in the binary Willshaw model is constructed (i.e. it contains fully connected neural assemblies). There is an additional delay matrix d which gives delays of the autoassociation in units of the relative time scale. In our implementation, these delays are however all set to zero. Then, the autoassociative spike counter for a global step s is given by $c_i^A(t)=\sum_{j=1}^{N} y_i^s(t-d_{ji})a_{ji}$, where i and j are neuron indices. The autoassociative spike counter for neuron i increases when many neurons that are connected to neuron i have already fired within this global step. The population activity counter for global step s is defined by $c_i^\Sigma(t)=\sum_{j\in P(i)} y_j^s(t)$. It counts how many neurons in the whole cortical module have already fired. Note that $c_i^\Sigma(t)\geq c_i^A(t)$.

The different cortical modules within the model are connected via heteroassociative coupling matrices that map a pattern in the source module to a corresponding pattern in the target module. For simplicity, we allow only one heteroassociative matrix between the same two cortical modules, in particular this prohibits several heteroassociative connections between the same two modules with different delay values. Assume that our network consists of M cortical modules, then, the heteroassociative matrix from module p to module q is called H^{pq}. It has an associated delay matrix d^{pq} which gives delays in relative time units and a global delay value D^{pq} which gives the delay of the whole heteroassociation in global time steps. Note that in our model $D^{pq}\geq 1$ for all heteroasso-

ciations. Then we can define the heteroassociative spike counter of the cortical module p as $c_i^H(s,t)=\Theta\sum_q r(s-D_{\langle i\rangle}^{pq},t-d_{\langle i\rangle}^{pq})\,H^{pq}$, where $D_{\langle i\rangle}$ is the i-th column of matrix D. The heteroassociative spike counter gives the number of neurons from which the current neuron i gets input, weighted with the instantaneous rate r_i, from how many neurons the current neuron i gets input. The value $c_i^H(s,\infty)$ represents the total amount of heteroassociative input that the neuron is addressed with, regardless of the relative spike times.

There is an additional feedback spike counter defined by $c_i^F(s,t)=\Theta\cdot r(s-1,t)\cdot A$, where $r(s-1,t)$ is simply the row vector of all instantaneous rates in the current module. The feedback counter is an autoassociation from one global time step to the next and is used to keep patterns active over several global time steps.

With the spike counter variables, the membrane potential for a neuron within a cortical module p can now be defined as

$$\dot{x}(t)=a\cdot c^H(s,t)+b\cdot c^H(s,\infty)\cdot\alpha\cdot\left(c+L\left(\frac{c^A(s,t)}{c^\Sigma(s,t)}\right)\right)+d\cdot c^F(s,t)+e\cdot c^F(s,\infty),$$

where $x(0)=0$.

Here a,b,c,d and e are weighting factors, typical values are a=0.02, b=-50, c=0. The variables d and e are zero for modules with no memory over time and positive for modules with memory, where the memory effect becomes stronger for high values of d and e.

The function L can be thought of as something between logarithmic and linear function, in our current implementation, we use the identity.

The parameter α represents the separation strength and is introduced just for convenience and analogy with the original spike counter model introduced in [29], in our model, the same effect could as well be achieved with the parameter b. In contrast to the model parameters a,b,c,d and e, the value of α can be changed during runtime. For example, it is influenced by the feedback connection from the quality measure to the separation strength.

After a neuron spiked for the first time, its membrane potential is reset to zero. More than two spikes per neuron are not allowed within one global time step. The second spike is completely ignored by most parts of the model, it is only used for calculation of the quality measure (see below). It is easy to see that the algorithm terminates in each global time step: either all neurons have spiked twice or at some point, all neurons that are still allowed to spike have non-positive derivative of their membrane potential and no more changes to the membrane potential

are possible (this happens after all heteroassociative inputs for that global step have been taken into consideration).

8.2 Quality Measure

For the estimation of the retrieval quality, we actually compare two retrievals, one with high fixed value of the separation strength α and one with the current value based on the last quality measure. The first retrieval with high separation strength α will be referred to as "control retrieval".

The quality measure is initialised as 0.9^k, where k is the number of neurons that either spiked in the control retrieval but did not spike twice in the second retrieval or that spiked twice in the second retrieval but did not spike at all in the control retrieval. It is then multiplied with $0{,}95+0{,}05\cdot\prod\frac{c_i^A(s,t)}{c_i^\Sigma(s,t)}$, where the product is over all neurons that emitted a spike in the second retrieval. An analogous term $\prod\frac{\tilde{c}_i^A(s,t)}{\tilde{c}_i^\Sigma(s,t)}$ of the control retrieval is then multiplied by the quality measure, where $\tilde{c}_i^X$ refers to the parameters of the control retrieval. The ratio from the control retrieval has stronger weight because there, only one assembly is able to spike (the one which is addressed strongest) and thus the product is a strong measure for the address quality. There is an additional punishment factor if the number of active neurons in the second retrieval is more than 20% away from the ideal pattern size, the factor is then $\begin{cases} n/k+0{,}2 & \text{if } n/k<0{,}8 \\ k/n+0{,}2 & \text{if } k/n<0{,}8 \\ 1 & \text{otherwise} \end{cases}$ where n is the number of active neurons in the second retrieval and k is the ideal pattern size.

8.3 Online Learning

During online learning, new patterns are generated when the quality measure becomes too low (the threshold is currently set to 0,95) or if a population gets a signal that motivates it to learn a new pattern regardless of its own quality measure. The second mechanism is required if e.g. a sensory input population P1 is heteroassociatively connected to a second population P2 and P2 basically encodes the same information as P1 and is mainly driven by P1, but might e.g. get some weak additional context information. Then, whenever a new representation in P1

is generated, a new representation in P2 is also required. This does not happen automatically if the new pattern in P1 is similar to an old one, because the heteroassociation between P1 and P2 might "correct" that and P2 will activate the similar old pattern. Thus, P1 can motivate P2 to generate a new pattern if it created a new pattern itself by sending an additional special learn signal to P2.

Let k be the ideal size of patterns for the population under consideration. Whenever new patterns are generated, $r \cdot k$ neurons are chosen randomly and up to $(1-r) \cdot k$ are chosen from the strongest activated neurons in the population. Here, $r \in [0,1]$ is a parameter that controls the randomisation of the patterns. For sensory input areas, no random neurons should be added, while in other cases, randomisation of the patterns might help to avoid large pattern overlap possibly caused by the heteroassociative input matrix. Ideal pattern size might not be reached when there are not enough neurons receiving strong enough input.

The heteroassociations between the populations are updated via binary Hebbian learning, i.e. whenever the target and source population of a heteroassociation have active patterns and either of them is new, the heteroassociative matrix is updated accordingly with additional one-entries.

8.4 Sequence Memory

In the following we give a short description how our cortical modules can be used to represent sequences as it happens in area A4 in the cortical language model.

To represent a sequence of patterns, first, the patterns are stored autoassociatively in the cortical module. Then, the pattern transitions corresponding to the sequence are stored heteroassociatively in a connection from the cortical module onto itself. To make this unique, the patterns are not allowed to appear twice in a sequence (e.g. the sequence "A B C A D" would not be valid). This however can be easily resolved by adding additional discriminating neurons to the patterns of the elements that appear several times (e.g. "Aa B C Ab D" in the previous example, where "Aa" and "Ab" have large overlap, but still have some discriminating neurons). The heteroassociation needs to be weaker than the autoassociative feedback and also it must have greater delay. Then, whenever a pattern is active in the cortical module and no further input is applied, the pattern stays active because of the strong autoassociation. To switch to the next pattern, the whole module needs to be inhibited for a time shorter than the delay of the heteroassociation, but longer than the delay of the autoassociation). After releasing the inhibition, the autoassociation is not effective any more, and the heteroassociation switches the cortical module to the next state in the sequence.

In our current model, sequence memories are not learnt or extended during performance. Instead, the patterns as well as all auto- and heteroassociative connections are constructed when the network is initialised and are then kept fix throughout the simulation.

9 References

[1] Hebb, D.O. *The organization of behavior. A neuropsychological theory.* Wiley, New York 1949

[2] Braitenberg, V. Cell assemblies in the cerebral cortex. In Heim, R. and Palm, G., Eds. *Lecture notes in biomathematics (21). Theoretical approaches to complex systems.*, Springer, Berlin Heidelberg New York, 1978

[3] Palm, G. On associative memories. *Biological Cybernetics* 36 19-31, 1980

[4] Palm, G. *Neural Assemblies. An Alternative Approach to Artificial Intelligence.* Springer, Berlin 1982

[5] Palm, G. Associative networks and cell assemblies. In Palm, G. and Aertsen, A., Eds. *Brain Theory*, Springer, Berlin Heidelberg, 1986

[6] Braitenberg, V. and Schüz, A. *Anatomy of the cortex. Statistics and geometry.* Springer-Verlag, Berlin 1991

[7] Palm, G. Cell assemblies as a guideline for brain research. *Concepts in Neuroscience* 1 133-148, 1990

[8] Palm, G. Memory capacities of local rules for synaptic modification. A comparative review. *Concepts in Neuroscience* 2 97-128, 1991

[9] Palm, G. Computing with neural networks. *Science* 235 1227-1228, 1987

[10] Braitenberg, V. *On the texture of brains* Springer, Berlin Heidelberg New York 1977

[11] Palm, G. Rules for Synaptic Changes and their Relevance for the Storage of Information in the Brain *Cybernetics and Systems Research* 1982

[12] von der Malsburg, C. Am I thinking assemblies? In Palm, G. and Aertsen, A., Eds. *Brain Theory*, Springer, Berlin Heidelberg, 1986

[13] Singer, W. and Gray, C.M. Visual feature integration and the temporal correlation hypothesis. *Annu.Rev.Neurosci.* 18 555-586, 1995

[14] Eckhorn, R. and Bauer, R. and Jordan, W. and Brosch, M. and Kruse, W. and Munk, M. and Reitboeck, H.J. Coherent Oscillations: A mechanism of feature linking in the visual cortex? *Biol. Cybern.* 60 121-130, 1988

[15] Ajjanagadde, V., Shastri, L. Rules and variables in neural nets *Neural Computation* 3 121-134, 1991

[16] Bienenstock, E. Composition In Aertsen A., Braitenberg V., Eds. *Brain Theory. Biological basis and computational principles*, Elsevier, Amsterdam, 1996

[17] Knoblauch, A. and Palm, G. Pattern separation and synchronization in spiking associative memories and visual areas. *Neural Networks* 14 763-780, 2001

[18] Knoblauch, A. and Palm, G. Scene segmentation by spike synchronization in reciprocally connected visual areas. II. Global assemblies and synchronization on larger space and time scales. *Biological Cybernetics* 87(3) 168-184, 2002

[19] Abeles, M. *Corticonics: Neural circuits of the cerebral cortex.* Cambridge University Press, Cambridge UK 1991

[20] Diesmann, M. and Gewaltig, M.O. and Aertsen, A. Stable propagation of synchronous spiking in cortical neural networks. *Nature* 402(6761) 529-533, 1999

[21] Abeles, M. and Bergman, H. and Margalit, E. and Vaadia, E. Spatio-temporal firing patterns in frontal cortex of behaving monkeys. *Journal of Neurophysiology* 70 1629-1643, 1993

[22] Pulvermüller, F. Sequence detectors as a basis of grammar in the brain. *Theory in Bioscience* 122 87-103, 2003

[23] Pulvermüller, F. *The neuroscience of language: on brain circuits of words and serial order.* Cambridge University Press, Cambridge, UK 2002

[24] Pulvermüller, F. Words in the brain's language. *Behavioral and Brain Sciences* 22 253-336, 1999

[25] Y. Ikegaya, G. Aaron, R. Cossart, D. Aronov, I. Lampl, D. Ferster,R. Yuste, Synfire chains and cortical songs: temporal modules ofcortical activity *Science* 304 559–564,

[26] Lin, L. and Osan, R. and Tsien, J.Z. Organizing principles of real-time memory encoding: neural clique assemblies and universal neural codes *Trends in Neurosciences* 29(1) 48-57, 2006

[27] Fay, R. and Kaufmann, U. and Knoblauch, A. and Markert, H. and Palm, G. Combining Visual Attention, Object Recognition and Associative Information Processing in a NeuroBotic System In Wermter, S. and Palm, G. and Elshaw, M., Eds. *Biomimetic Neural Learning for Intelligent Robots*, Springer, Berlin Heidelberg New York, 2004

[28] Knoblauch, A. and Markert, H. and Palm, G. An associative cortical model of language understanding and action planning In Mira, J. and Alvarez, J.R., Eds. *Artificial Intelligence and Knowledge Engineering Applications: A Bioinspired Approach*, Springer, Berlin Heidelberg New York, 2005

[29] Knoblauch, A. Synchronization and pattern separation in spiking associative memory and visual cortical areas. *PhD thesis, Department of Neural Information Processing, University of Ulm, Germany* 2003

[30] Willshaw, D.J. and Buneman, O.P. and Longuet-Higgins, H.C. Non-holographic associative memory. *Nature* 222 960-962, 1969

[31] Palm, G. Local Rules for Synaptic Modification in Neural Networks *Journal of Computational Neuroscience* 1990

[32] Palm, G. On the Information Storage Capacity of Local Learning *Neural Computation* 4 703-711, 1992

[33] Palm, G. and Sommer, F.T. Information capacity in recurrent McCulloch-Pitts networks with sparsely coded memory states *Network* 3 177-186, 1992

[34] Lapicque, L. Recherches quantitatives sur l'excitation electrique des nerfs traitee comme une polarization *Journal de Physiologie et Pathologie Général* 9 620-635, 1907

[35] Dayan, P., Abbot, L.F. *Theoretical Neuroscience* MIT Press, Cambridge, Massachusetts 2001

[36] Markert, H., Knoblauch, A. and Palm, G. Associative Language Processing in Cortical Areas *Proceedings of the IEEE SMC UK-RI Chapter Conference 2005 on Applied Cybernetics*, 2005

[37] Rabiner, L., Juang, B.H. *Fundamentals of Speech Recognition* Prentice Hall, Englewood Cliffs, N.J. 1993

[38] Rabiner, L. A tutorial on hidden Markov models and selected applications in speech recognition *Proc. IEEE* 77 257-285, 1989

[39] Knoblauch, A., Fay, R., Kaufmann, U., Markert, H. and Palm, G. Associating words to visually recognized objects. In Coradeschi, S. and Saffiotti, A., Eds. *Anchoring symbols to sensor data. Papers from the AAAI Workshop. Technical Report WS-04-03*, AAAI Press, Menlo Park, California, 2004

[40] Markert, H., Knoblauch, A. and Palm, G. Detecting Sequences and Understanding Language with Neural Associative Memories and Cell Assemblies In Wermter, S., Palm, G. and Elshaw, M., Eds. *Biomimetic Neural Learning for Intelligent Robots*, Springer, Heidelberg, New York, 2005

[41] Siegelmann, H.T., Giles, C.L. The Complexity of Language Recognition by Neural Networks *Neurocomputing* 15 327-345, 1997

[42] Maass, W., Sontag, E. Analog Neural Nets with Gaussian or other Common Noise Distribution cannot Recognize Arbitrary Regular Languages *Neural Computation* 11 771-782, 1999

[43] Arbib, M.A. and Billard, A. and Iacoboni, M. and Oztop, E. Synthetic Brain Imaging: Grasping, mirror neurons and imitation. *Neural Networks* 13(8-9) 975-997, 2000

[44] Arbib, M.A. and Fagg, A.H. and Grafton, S.T. Synthetic PET imaging for grasping: From primate neurophysiology to human behavior. In Sommer, F.T., Wichert, A., Eds. *Exploratory Analysis and Data Modelling in Functional Neuroimaging*, MIT Press, Boston, MA, 2002

[45] McKinstry, J.L., Edelman, G.M., Kirchmar, J.L. A cerebellar model for predictive motor control tested in a brain-based device *Proc Natl Acad Sci USA* 103 3387-3392, 2006

[46] Marr, D. *Vision.* W.H.Freeman, New York 1982

[47] Somers, D.C. and Nelson, S.B. and Sur, M. An emergent model of orientation selectivity in cat visual cortical simple cells. *J.Neurosci.* 15 5448-5465, 1995

[48] Grossberg, S. and Pessoa, L. Texture segregation, surface representation, and figure-ground separation. *Vision Research* 38 2657-2684, 1998

[49] Arbib, M.A. *The Handbook of Brain Theory and Neural Networks, Second Edition* MIT Press, Cambridge, MA 2003

[50] O'Reilly, R.C. and Braver, T.S. and Cohen, J.D. A biologically based computational model of working memory. In Miyake, A. and Shah, P., Eds. *Models of working memory: Mechanisms of active maintenance and executive control*, Cambridge University Press, New York, 1999

[51] Horwitz, B. and Friston, K.J. and Taylor, J.G. Neural modeling and functional brain imaging: an overview. *Neural Networks* 13 829-846, 2000

[52] Nilsson, N.J. *Artificial Intelligence: A New Synthesis* Morgan Kaufmann, San Francisco 1998

[53] Fodor, J.A. and Pylyshyn, Z.W. Connectionism and cognitive architecture: A critical analysis *Cognition* 28 3-71, 1988

[54] Sun, R. and Bookman, L.A. *Computational Architectures Integrating Neural and Symbolic Processing* Kluwer Academic Publishers, 1995

[55] Rumelhart, D. and McClelland, J. *Parallel distributed processing.*. MIT Press, Cambridge, MA 1986

[56] Shastri, L. and Ajjanagadde, V. From simple associations to systematic reasoning: a connectionistic representation of rules, variables and dynamic bindings. *Behavioral and Brain Sciences* 16(3) 417-494, 1993

[57] Harnad, S. The symbol grounding problem)e, *Physica D* 42 335-346, 1990

[58] Giles, C.L., Omlin, Ch.W. Extraction, Insertion and Refinement of Symbolic Rules in Dynamically Driven Recurrent Networks *Connection Science* 5 307-337, 1993

[59] Goonatikale, S., Khebbal, S. *Intelligent Hybrid Systems* John Wiley & Sons Ltd, 1995

[60] Elman, J.L. Finding structure in time. *Cognitive Science* 14 179-211, 1990

[61] Wermter, S. *Hybrid Connectionist Natural Language Processing* Chapman and Hall, London 1995

[62] Miikkulainen, R., Bijwaard, D. Parsing Embedded Clauses with Distributed Neural Networks In *Proceedings of the twelfth National Conference on AI (AAAI '94)*, AAAI Press, MIT Press, Menlo Park, CAL 1994

[63] Gori, M., Maggini, M., Soda, G. Learning Regular Grammars From Noisy Examples Using Recurrent Neural Networks *Technical Report* Dipartimento di Sistemi e Informatica, Universita di Firenze , 1995

[64] Steels, L. and Brooks, R. *The artificial life route to artificial intelligence: Building embodied, situated agents.* Erlbaum, Hillsdale, NJ 1995

[65] Talmy, L. *Toward a cognitive semantics: Vol. I. Conceptual structuring systems.* MIT Press, Cambridge, MA 2000

[66] Brooks, R.A., Stein, L.A. Building Brains for Bodies *Autonomous Robots* 1 7-25, 1994

[67] Hopcroft, J. and Ullman, J. *Formal languages and their relation to automata.* Addison-Wesley, Reading, Massachusetts 1969

[68] Alon, N., Dewdney, A., Ott, T.J. Efficient Simulation of Finite Automata by Neural Nets *Journal of the ACM* 38(2) 495-514, 1991

[69] Alquézar, R., Sanfeliu, A. An Algebraic Framework to Represent Finite State Machines in Single-Layer Recurrent Networks *Neural Computation* 7(5) 931-949, 1996

[70] Frasconi, P., Gori, M., Maggini, M., Soda, G. Representation of Finite State automata in Recurrent Radial Basis Function Networks *Machine Learning* 23(1) 5-32, 1996

[71] Horne, B.G., Hush, D.R. Bounds on the Complexity of Recurrent Neural Network Implementations of Finite State Machines *Neural Networks* 9(2) 243-252, 1996

[72] Gori, M. and Küchler, A. and Sperduti, A. On the implementation of frontier-to-root tree automata in recursive neural networks *IEEE Transactions on Neural Networks* 10(6) 1305-1314, 1999

[73] Giles, L., Gori, M. *Adaptive Processing of Sequences and Data Structures* Springer, Heidelberg, New York 1998

[74] Krichmar, J.L. and Edelman, G. Machine psychology: Autonomous behavior, perceptual categorization and conditioning in a brain-based device. *Cerebral Cortex* 12(8) 818-830, 2002

[75] Roy, D. Learning visually grounded words and syntax for a scene description task. *Computer Speech and Language* 16(3) 353-385, 2002

[76] Billard, A. and Hayes, G. DRAMA, a connectionist architecture for control and learning in autonomous robots. *Adaptive Behavior Journal* 7(1) 35-64, 1999