

AN ARCHITECTURE FOR DISTRIBUTED VISUAL MEMORY

Nils Jungclaus, Markus von der Heyde, Helge Ritter, Gerhard Sagerer

University of Bielefeld, SFB 360*

Postfach 100131, 33501 Bielefeld, Germany

email: nils@techfak.uni-bielefeld.de

Abstract

The development of autonomous as well as situated robots is one of the great remaining challenges and involves a number of different scientific disciplines. In spite of recent dramatic progress, it remains worthwhile to examine natural systems, because their abilities are still out of reach. Motivated by research work done in the fields of cognitive systems, visual perception, and psychology of memory we designed and implemented a memory architecture for visual tasks. Structural and functional concepts of the memory architecture were modeled on the ones found in natural systems. We present an efficient implementation based on parallel programming techniques.

The memory module is integrated into a distributed system for speech and image analysis, which is currently developed in the *Sonderforschungsbereich (SFB) 360, Situated Artificial Communicators*, where a hybrid vision system combining neural and semantic networks is used.

1 Introduction

The presence of a stable world with well-identifiable objects is one of the most pervasive impressions generated by our brain. Its assembly from the sensory signals of our perceptual apparatus is an amazing feat: looking at the nerve signals delivered from our sensors to the brain hardly suggests that they signal much about any well-defined and stable entities around us.

A very similar experience is shared by the engineer that tries to construct an intelligent machine that is able to sense its environment, to detect objects and to guide its actions to make intelligent-seeming, goal-directed behavior possible. Again, it turns out that it is extremely hard to transform the video signals of a camera or the voltage of a microphone into something that correlates well with the natural entities by which our brain builds our world.

* This research was supported by the German Research Foundation (DFG) within SFB 360. Only the authors are responsible for the contents of this publication.

In both cases, we are confronted with the question, what kind of processing may allow us to transform the the bewildering variability of sensory signals – be they natural or artificial – into more stable entities that correlate reliably and stably with what we perceive.

There is little doubt that various forms of *memory* are key elements in this process. The experience of *stability* requires *re-cognition* and *familiarity*, which are all concepts that are by necessity based on some memory mechanism. On shorter time scales, the impression of *object constancy*, e.g., during a temporary occlusion, provides an example of another memory function that starts to appear in the human infant only a few months after birth. On even shorter time scales, the perception of continuous motion and our ability to predict its imminent course can be viewed as a form of a short time memory specialized for this type of task. Finally, all kinds of learning processes, ranging from the formation of simple conditioned reflexes over the chaining of elementary motor programs to the elaboration of complex systems of associatively interrelated concepts are all based in an essential way on some type of memory mechanism.

This very coarse list may suffice to indicate that memory is not a single, uniform capability, but comprises a number of possibly related mechanisms operating at several levels of abstraction and on several time scales. From the abstract viewpoint of an engineer, however, we may discern a common, shared goal of all these processes: they *bind together* or “*associate*” *different pieces of information*, either across a spatial domain (leading to recognizable *spatial patterns* suggesting themselves as “objects”), or across *time*, leading to the recognition of the familiar and to the possibility of such fundamental concepts as *causality*, which are most basic for our ordering of the world.

In view of this shared goal of different memory mechanisms, it might be useful to compare engineering approaches with insights about the structure of biological memory mechanisms in the brain. The benefit of such an approach may be mutual: as engineers, we may get important architectural clues as how to structure our artificial systems; for the biology side, the attempt to implement concepts about biological memory mechanisms on a computer may shed light on the workability of ideas and may help to focus new questions.

The present paper aims to make a contribution along these lines. Based on the experience of a large scale research effort in the context of the Bielefeld Sonderforschungsbereich 360 aiming to build an artificial system that is able to perceive its surroundings while executing actions and communicating with a human instructor, we describe an approach of how to implement a distributed memory mechanism for such a system, focusing on the domain of vision and borrowing some organizational features from the visual system in the brain.

In the following, we will first refer to some results showing how a technical memory architecture could be organized and then, taking the environment of the SFB system into account, present

an approach for a generic memory architecture for the visual subsystem.

2 Hierarchical Organization of Memory

The intelligent behavior of natural systems is related to the physical world. All kinds of behavior are based on experiences and the sensation, perception, and interpretation of certain aspects of their environment. It is widely agreed that for such complex tasks lower-level neural processes cooperate with higher-level symbolic programs. Additionally, the required system complexity forces a hierarchical organization to achieve stability and robustness (Simon 1962). For cognitive systems Newell (1990) proposed a hierarchy of abstraction levels, and postulated that more abstract primitives describe larger units in space and/or time and need more time to be computed or executed. Ballard (Ballard, Hayhoe, and Pook 1995; Ballard 1997) elaborated these ideas further and refined the time scales using the results of experimental observations. Furthermore, even separate processes running under different time scales are integrated in the perception-action cycle. Ballard points out that “Cognitive and perceptual processes cannot be separated, and in fact are interlocked for reasons of computational economy”.

One major part of the perceptual and cognitive apparatus is the visual system. A model of the hierarchies and the parallel visual pathways is described by DeYoe and van Essen (1988). Treisman (1988) developed a model of the stages in visual perception. Based on experiments, the features color, orientation, size, and distance are described. There is significant evidence that the visual system uses a distributed memory encoding different types of features like those postulated by Treisman, as well as structures of different complexity according to the hierarchies as postulated by van Essen. Different aspects of memory have already been described in literature (Baddeley 1976). Kanerva (1988) developed a sparse distributed coding scheme. An iconic representation based on steerable filters is described in (Rao and Ballard 1995; Ballard 1997). Motivated by the different types and hierarchies of representation in the visual pathway we will describe a uniform architecture for a distributed visual memory. The development is motivated both by a neuroscientific background and by an application within a system for man-machine-interaction.

3 Memory Functions in Vision

In the SFB, a long term research project involving researchers in the field of pattern recognition, artificial intelligence, and linguists, one major goal is the development of a prototype for an artificial situated communicator. The SFB-Demonstration-System is an integrated system

for speech and image understanding, acting in a situated domain where a toy object, such as a “plane” made of wooden pieces like screws, cubes or bars, is to be assembled by the instructions of a human operator in a cooperative way (see p.e. Fink, Jungclauss, Kummert, Ritter, and Sagerer (1996)). Currently, the system is able to carry out a dialog about referencing an object in a scene of several non-overlapping objects. It is not intended to become an autonomous system that is able to act in some kind of environment by itself; rather we focus on natural ways of interaction with an artificial system.

To achieve such an ability requires to integrate a considerable number of different modules that contribute partial capabilities, such as the recognition and location of objects, scene interpretation, speech recognition and interpretation, the planning of actions and the planning of the conversation.

To coordinate the operations of these modules, we coarsely adopt an architecture of parallel visual pathways as it is known to exist in the brains of higher animals, such as the macaque (DeYoe and van Essen 1988). In our hybrid vision architecture (Figure 1), we use implemented several processing pathways:

- The perceptual grouping of contours (Maßmann and Posch 1995) is carried out in parallel on the left and right channel of intensity images.
- Color images are used by a holistic object recognition module that is based on neural networks using features retrieved from optimized Gabor filters (Heidemann and Ritter 1996).
- In parallel, a polynomial classifier is used to segment the color images into regions and some basic features of these regions are computed.

The results of the last two processing streams are evaluated in a knowledge-based object recognition module. Based on the initial hypotheses of the holistic object recognition, regions are combined by a semantic network to provide new verified object hypotheses (Heidemann, Kummert, Ritter, and Sagerer 1996; Sagerer and Niemann 1997).

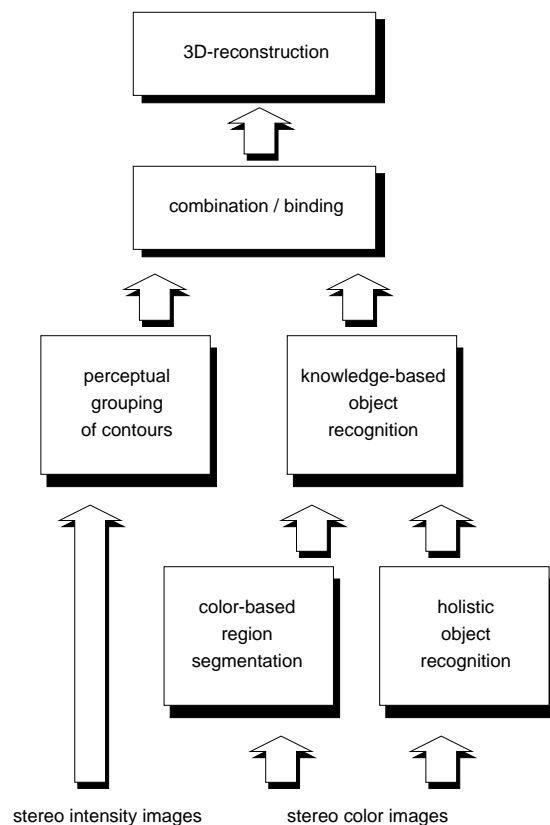


Figure 1: Hybrid Vision Architecture

Subsequent processing steps combine contours from the perceptive grouping pathway with the object hypotheses obtained from the fusion of the last two processing pathways to achieve a 3D-reconstruction of the scene. By now, the computation in our prototype architecture is strictly data-driven or bottom-up. Improvements may be achieved by using results on higher levels as expectations for lower level algorithms which introduces a model-driven top-down evaluation. Ideally, both types of computation should be done concurrently which requires some kind of synchronization in points where these evaluations meet.

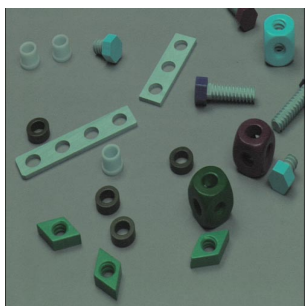
Most image processing algorithms work on single images, not considering the development of a scene over time. This raises problems when references to objects occur that are not resolvable using the current view of the scene alone, e.g. in situations like the ones shown below. Some kind of memory function is necessary to fulfill this task.



A typical scene in our construction scenario contains several parts of a wooden child's toy toolkit. As long as the scene is static and not disturbed by intervening actuators like human or robotic arms, the vision system is able to recognize about 95% of the objects correctly.



However, if there occurs some action in the scene, the vision system alone is not able to deliver enough information for a dialog about it. In this example, there is a hand hiding some of the parts in the upper left corner, making them nonexistent for the vision system if they are not memorized from previous images. A second effect is caused by changing light conditions (shadow) introduced by the hand: The measured colors of the objects become changed which might confuse the recognition algorithms. Here, memorized knowledge about the scene can be used to stabilize recognition over time using other features but color of the object.



Moved objects like the two cubes on the right in this scene (in comparison to the first example) cause similar problems when talking about the scene. We should know that the objects are the same as before since only their location changed but other features like the type of the object or the color are still the same. Again, a memory based association would help handling the situation.

Conventional technical solutions that store information over time like typical databases are not particularly well suited for the given task since we need at least two mechanisms that are usually not covered by them: *(i)* we need some kind of association to connect stored and new information and *(ii)* due to limited capacity of computers we need some automatic deletion, respectively some kind of forgetting in more natural terms.

In the following section we present an architecture for a universal memory module that addresses these issues. Its integration with the existing system is achieved by extending the functionality of the arrows between the modules shown in Fig. 1 resulting in a modified architecture that is partially shown in Fig. 2. The output of the modules is passed to the memory instead of being sent to the following module. The memory module stores incoming data over time as described in the following section and passes only changes to the following module. Additional interfaces are provided for top-down access to the stored information.

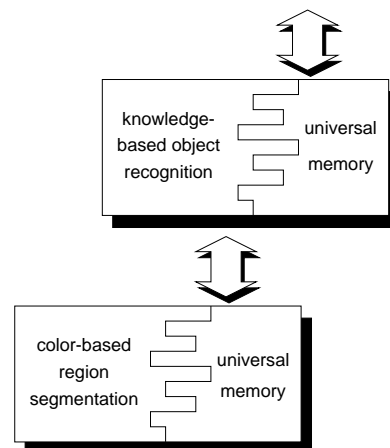


Figure 2: A Part of the Architecture extended with Memory

4 A Proposal for a Generic Memory Architecture

A pragmatic approach to achieve the desired capabilities of our system might have consisted in adding suitably specialized memory functionality in all places where needed.

However, since we are not only interested in building a working demonstrator, but also to learn about principles that can generally aid the construction of such systems, we were tempted to try a more ambitious approach and to attempt the development of a rather generic memory architecture that is sufficiently general to be used at a variety of processing levels, requiring at most minor modifications to adapt it to its specific task.

For the functionality of this memory architecture, we had the following three primary goals in mind:

1. The memory architecture should support a distributed storage of the attribute values that characterize instances of previously encountered objects. To achieve this, we were guided by the storage of object features in topographic feature maps in the brain and chose a representation based on several hierarchical self-organizing maps, each map being assigned to cover a particular feature subspace.

2. Based on this storage mechanism, the memory should support an associative completion of partially specified feature conjunctions. The completion should be based on the statistics of previously seen objects, returning “typical” values for the filled in attributes.
3. The associative completion mechanism should be useful for the “smoothing” of data streams. In this mode, the memory is used as a “filter” that is able to correct minor recognition errors or to temporarily substitute missing values, resulting, e.g. from temporary occlusions of objects or sudden changes in illumination. In this role, the memory architecture makes an essential contribution to the continuous maintenance of a stable world model.

In view of the attempted genericity, a further important requirement was a realization of these functionalities that supports their flexible embedding at many different levels in the processing hierarchy of the demonstrator system. To achieve this, we implemented the memory in close interaction with the existing software system DACS (Fink, Jungclaus, Ritter, and Sagerer 1995) that is used to establish the communication between all processing modules. The localization of memory in a relatively low level directly attached to communication links in contrast to a single highlevel module where all knowledge is collected seems as well to be consistent with our current view of storage mechanisms in the brain. By the embedding of the memory in the communication infrastructure, it has the required, intimate access to all data flow, allowing a natural realization of the goals 1-3 without any need to make changes to the implementation of the existing modules that use the communication layer for their data exchange.

The design of the external interface of our memory module is therefore primarily influenced by the current vision architecture of our prototype system. However, internal aspects like mechanisms for association or forgetting benefit extensively from knowledge about natural systems.

5 Associative Feature Storage

In our vision architecture, the modules continuously exchange data describing the current scene at different levels of abstraction. The color segmentation module e.g. produces regions of similarly colored pixels in the image. The data-structure describing such a region includes features like color, size, main direction, eccentricity or a surrounding polygon. Other modules exchange structures describing objects at some higher level including lists of contour line segments or even 3D-attributes. At a lower level, the used features may have less obvious interpretations, as, e.g., in the case of response amplitudes of Gabor filters used to coarsely mimic visual receptive fields in the “holistic” processing branch explained in the previous section 3.

On the technical side, when storing information in some kind of database, an important role

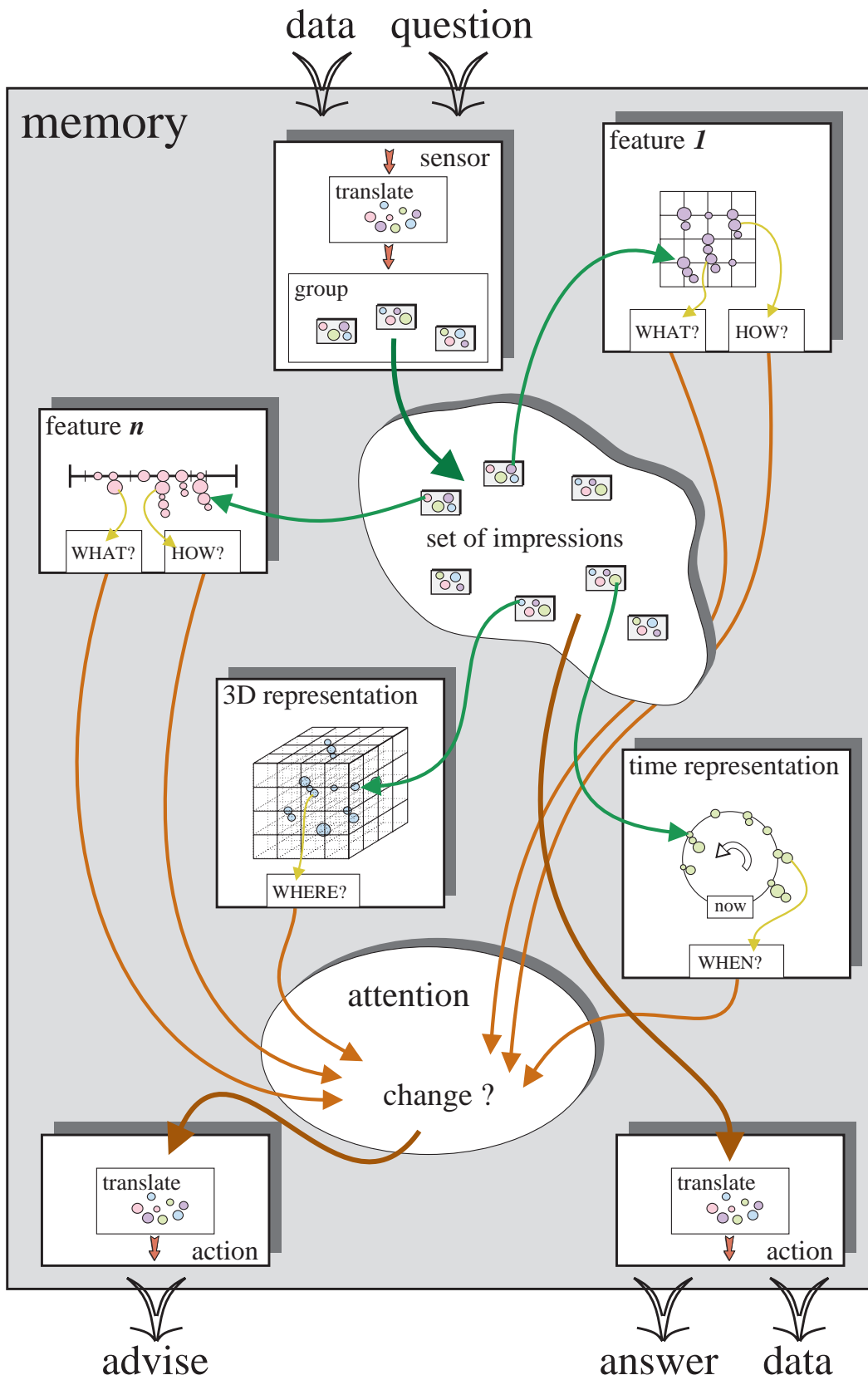


Figure 3: Information Flow inside the Memory (taken from Heyde (1997))

is played by a good indexing scheme supporting an efficient access to the data elements themselves. Ideally, a good indexing scheme should support queries that allow to retrieve data items according to their semantic similarity. However, the creation of such indices requires a-priori information about the data that often is not available. This makes it interesting to consider adaptive schemes that gradually learn data similarities from the statistics of the incoming data. One rather powerful scheme is to create “maps” that translate semantic similarities between data items into spatial neighborhood (Ritter and Kohonen 1989). Interestingly, such scheme also seems to play a major role in the organization of many areas in the cortex of the brain. In the visual cortex, e.g., we know of several different topographic maps which are usually retinotopically organized, with features such as color or ocularity represented as “secondary features” that give rise to columnar “stripe structures”. There are also more abstract maps, such as maps of auditory space in the superior colliculus that are aligned with a visual map, and there is some evidence, that also higher-level visual areas, such as the inferotemporal cortex, possess a topographic organization according to the similarity of high-level visual features (Fujita, Tanaka, Ito, and Kang 1990).

Consequently, we chose for our artificial memory system a separate artificial feature map for each feature dimension that we are interested in (currently, we restrict ourselves to a subset of the data elements of the incoming data structure, which are translated into a suitable format for internal use by the memory; this, however, is a technical point and will not be discussed further). In the following we will subsume the set of internally used features under an *impression* which should be understood in a technical sense. When a new object description is passed through the memory, those of its feature values for which the memory provides feature maps are put into the corresponding maps, indicated in Fig. 3 as major boxes. In view of the usually privileged roles of space and time, we provide two additional feature maps where each feature is stored: an associated 3D space map and an associated temporal map. These maps store the location in space and in time, resp., that belong to the feature with which the feature map was adapted. Only few of these links are shown in the figure to keep it comprehensible. The structure of the maps provides very efficient access to impressions with similar features which is used for the main functionality of the memory, the association.

After the transformation of the newly received data-structures to the internal data format, the association is done in two phases: (*i*) in each feature map we search for similar features already stored in the map, (*ii*) after retrieving the set of corresponding impressions, a statistical evaluation is done to determine which impressions share similar features in as many feature maps as possible.

The first phase of the association is directly supported by the storage structure of the feature maps. Depending on the type of feature we use different storage structures from simple linked

lists or rings up to hierarchical self organizing maps (SOM) (Kohonen 1995; Ritter, Martinetz and Schulten 1990) each of them holding hash arrays of fixed size at their nodes that provide a very efficient access to similar feature values. Similarity of features is determined by characteristic distance functions for each feature map. These functions have to be suitably specified according to the geometry of the input space.

Lists are primarily used to store features like classes of types or classified colors where a discrete range of feature values occurs. In this case the distance of two values usually is not meaningful or even not defined, we rather need a metric that takes the semantics of the values into account. Our memory architecture uses empirically determined conditional probability tables as a metric between symbol classes. Ring-lists are used for the time representation since they provide efficient access to the newest as well as the oldest entries which facilitates the implementation of forgetting mechanisms (section 7).

For all continuous feature spaces we use self organizing maps. Usually, these are employed to project a multidimensional input space to an output space of only few dimensions in a learning phase. The map may then be used as a predictor for some of the input dimensions or as a vector quantizer p.e. In our memory module, we have the need to store feature values of an a-priori unknown range and to provide an efficient access to them. Therefore, we use the SOMs in a slightly different way: The dimension of the SOM is chosen to be equal to the dimension of the input space to guarantee the preservation of neighborhood relations and the SOM works in learning mode all the time to achieve a continuous adaption to the distribution of the input data. Each node of the SOM has in addition to the usual prototype vector a fixed sized container that keeps a certain number of the most recent input values to its node. The prototype vector is adapted according to the standard Kohonen algorithm and represents a mean value of all features held in the container of the node. If a container is filled it may be replaced by a whole SOM, in which case its entries become distributed among newly created nodes. This way, hierarchical SOMs can be developed over time. developed over time.

In the second phase of the association, we extract a list of impressions based upon similarities found in the feature maps. A simple statistical evaluation is done to restrict the subset of returned impressions to those that are found to be similar in most feature maps. These are considered to be caused by the same object in reality. In this way, some of the features of an object may vary while the impressions are still connected to each other over time.

The described mechanism of forming associations is based on rather simple principles. Once the feature maps have been formed, their neighborhood structure defines the similarity criterion that underlies the formation of associations. Impressions are found to be similar by a joint evaluation of different feature maps after processing them independently. Therefore, the selection

of suitable internal storage structures influences the performance of the whole memory significantly. Beside the efficiency of data structures, the translation of the incoming data-structures into internal features and the design of the corresponding distance functions is most decisive for the functionality of the whole memory since only these are used in the association process.

6 Associative Retrieval and Filtering

There are mainly two output streams: One stream passes on those structures that could not be associated to already stored impressions. The presence of such non-matching data structures indicates “novelty” and is used to alert following modules to the presence of a possibly novel object or the occurrence of a significant change that may require a major re-evaluation of the current input.

The second stream is used for data structures for which some association with already stored impressions could be established. In this case, the memory will modify the attribute values of the original input in the direction of the average of the corresponding attributes of all those stored impressions that were associated with the input. In this way, we achieve a stabilization over time and are able to eliminate artifacts temporarily introduced into the scene, e.g., as a result of brief occlusions or due to sudden changes in illumination, as exemplified in section 3.

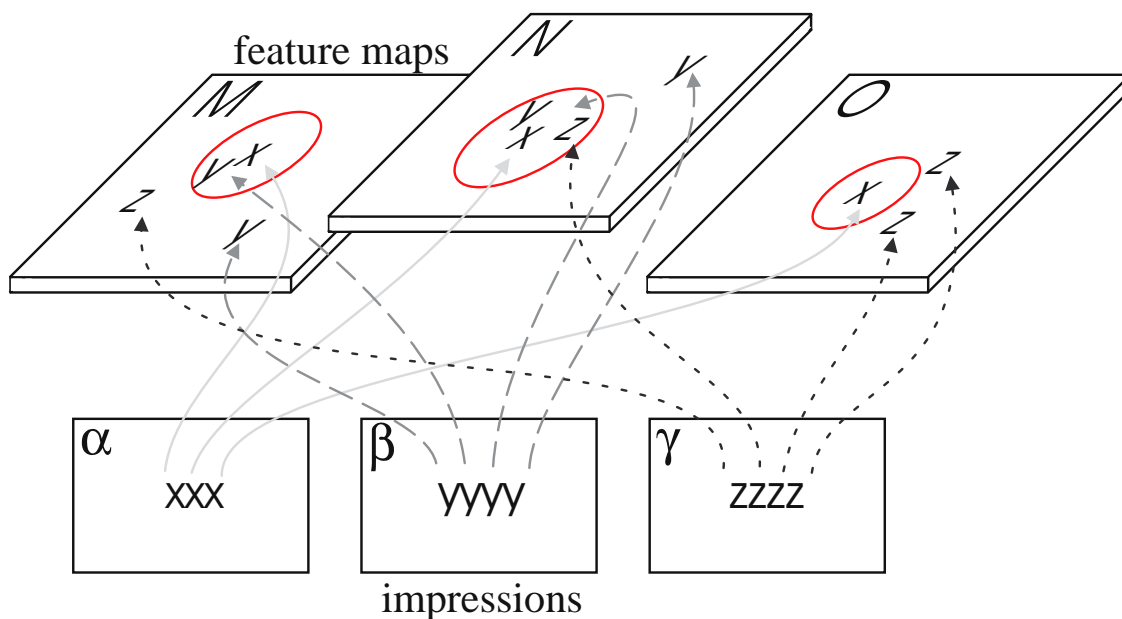


Figure 4: Association of Impressions

The association mechanism itself can be explained in more detail with the help of Fig. 4. Assume that impressions β and γ are already stored in the memory, and that α is an incoming, new

impression (“retrieval request”) for which we want to know whether it qualifies for becoming associated with one of the already stored impressions β and γ .

To indicate for each impression the set of measured feature values that characterize it, we employ letters x , y and z . E.g., the three x ’es in α might represent input values obtained for the features color, location and time. Each such feature value is sent (arrows) to the feature map that is responsible for the particular feature dimension (M for color, N for location and O for time; note that some impressions may contain several values for the same feature dimension, e.g., β might represent a two-colored object, with two y ’s going into the same (color-) feature map M).

Each feature map now allows to identify the subset of already stored feature values that lie within a neighborhood (indicated by circles) of the features from the current input. The outcome of this comparison is used to rank the already stored impressions according to their number of “hits” into the neighborhoods of the features of the current input impression (this ranking step is facilitated by keeping in our implementation in each feature map not only the feature values of each input, but also reference information to the data structure that describes the impression of which the feature value is part). Those impressions whose number of hits exceeds a given threshold finally become associated with the given input. In the present example, we find two “hits” for β , but only a single “hit” for γ and might, therefore return β as the only stored impression that is sufficiently similar to α to qualify for being associated.

We also can imitate some simple form of “imagination”: in this mode, the memory is fed with an internally generated set of “desired” feature values. The memory will then respond with a collection of impressions that match the specified feature values (have many “hits” in the neighborhoods of the specified values). This coarsely resembles our ability to imagine a previously seen picture when given some of its elements.

7 Forgetting

Forgetting is something a conventional database should never do. However, in our case the memory receives a stream of data-structures of theoretically unlimited length while the amount of storable information is limited by the hardware, so we are forced to keep only parts of it and provide a criteria to selectively delete other parts. Baddeley (1976) found two models to describe the process of forgetting that seem to work concurrently from psychological experiments. The one model assumes that we forget something caused by the overlay of new information, the other states that temporal effects are responsible for it. Both models are implemented in our memory module. Overlaying new information is done during storage where the limited size of

the hash tables in each of the nodes of the storing structures limits the number of similar values in a feature map.

Temporal forgetting is realized by an active process running in parallel. The feature map for time representation is searched for outdated entries periodically. The maximum storage time influences the character of the memory and may be used to simulate short or long term storage depending on the level of abstraction of the stored data-structures.

8 Discussion

We have implemented the described approach in the context of the SFB demonstrator scenario and tested it in the context of several situations, dealing with real world images involving temporary object occlusions, changes in lighting due to actions by the instructor, as well as scene changes caused by the addition or removal of objects.

The experience from these experiments shows that the presence of the memory module significantly helps to reduce or in some situations even completely remove several problems that occurred before in the vision sub-system of our demonstration-prototype for an artificial communicator. The memory delivers a stabilized representation of the scene resulting in a more robust dialog, even if in the presence of the described disturbances.

In addition, its ability to provide a flexible content-based access to previously stored information, the memory component tremendously facilitates the simultaneous bottom-up and top-down evaluation of results in the described the vision architecture. This leads to a marked increase in the efficiency of the overall system since in most cases only the fraction of data that has changed now needs to be passed on further, which considerable reduces the computational load in the following modules.

Figure 5 shows an example where regions of different color have become associated by virtue of the similarity of their other features. The left scene was presented to the memory for some time, then the input was switched to the scene on the right side.

The corresponding output of the memory is shown in Figure 6. The regions on the left image are the ones that are retrieved from the (actual) right image of Figure 5. The right image shows the regions associated to the actual ones. The bar on the left and the yellow screw on top did not change and are considered to be the same since all features are nearly the same. However, the cube and the screw on the right that changed color are associated as well since only one feature of the regions differs significantly compared to the left scene in Figure 5. Vanished parts like the orange rhomb nut at the bottom are still accessible from the memory using explicit requests.

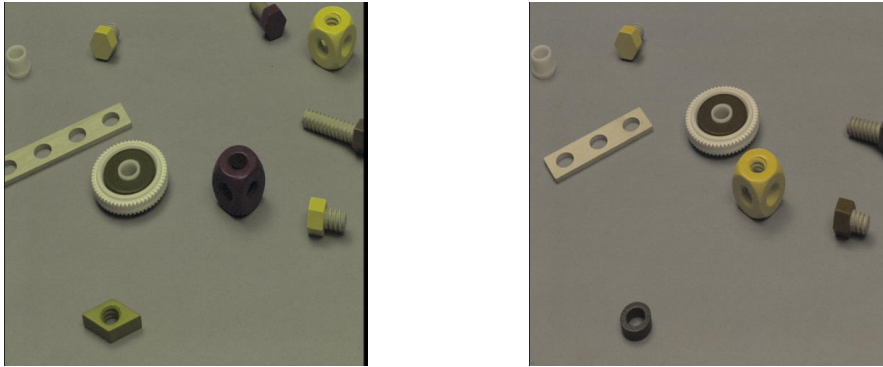


Figure 5: Two similar scenes used to provoke an association of objects with different colors (taken from Heyde (1997))

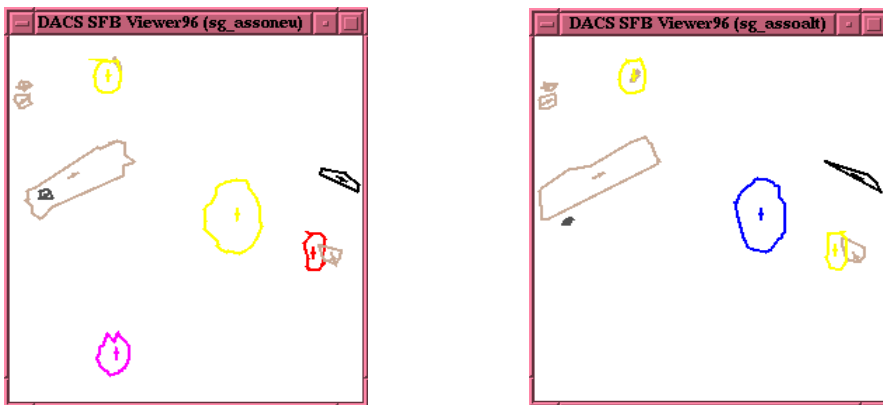


Figure 6: Output of the memory module showing the association of different colored objects of similar size at similar locations (taken from Heyde (1997))

We think that these experiences are rather encouraging with regard to translating architectural principles of biological vision systems in an abstracted form into a technical context and making them useful there. This is actually also one of the major goals of the SFB 360: to find ways to explore in a technical domain principles of information processing seen at the neural or at the cognitive levels in animals or people and to make them useful for practical applications.

Our future work will concentrate on the integration of the memory at the higher levels of our demonstration system. Objects will be recognized in 2D and 3D, and an internal scene representation will be provided by the memory module. A second aspect will be the integration of the memory into an active vision system to generate a representation of a whole scene while the camera delivers only partial images. A further goal will be the adaptive formation of concepts about patterns of spatio-temporal change, such as translation, rotation or various gestures conveyed by the human instructor.

We hope that taking this route will help us to contribute to a strengthening of the emerging bridge between modern computer science and current research into brain functions.

References

- Baddeley, A. D. (1976). *The psychology of memory* (1 ed.). New York: Harper & Row international.
- Ballard, D. H. (1997). *An Introduction to Natural Computation*. MIT Press.
- Ballard, D. H., M. M. Hayhoe, and P. K. Pook (1995). Diectic codes for the embodiment of cognition. Technical report, University of Rochester.
- DeYoe, E. A. and D. C. van Essen (1988). Concurrent processing streams in monkey visual cortex. *Trends Neurosci.* 11, 219–226.
- Fink, G. A., N. Jungclaus, F. Kummert, H. Ritter, and G. Sagerer (1996). A Distributed System for Integrated Speech and Image Understanding. In R. Soto (Ed.), *International Symposium on Artificial Intelligence*, Cancun, Mexico, pp. 117–126.
- Fink, G. A., N. Jungclaus, H. Ritter, and G. Sagerer (1995). A Communication Framework for Heterogeneous Distributed Pattern Analysis. In V. L. Narasimhan (Ed.), *International Conference on Algorithms and Applications for Parallel Processing*, Brisbane, Australia, pp. 881–890. IEEE.
- Fujita I., Tanaka K., Ito M., Kang C. (1992) *Columns for visual features of objects in monkey inferotemporal cortex*. Nature 360:343-346.
- Heidemann, G., F. Kummert, H. Ritter, and G. Sagerer (1996). A hybrid object recognition architecture. In *Proceedings of ICANN 96*. Springer Verlag.
- Heidemann, G. and H. Ritter (1996). A Neural 3-D Object Recognition Architecture Using Optimized Gabor Filters. In *Proceedings of 13th International Conference on Pattern Recognition, Vienna*, Volume IV, pp. 70–74. IEEE Computer Society Press.
- Kanerva, P. (1988). *Sparse Distributed Memory*. Bradford Books, Cambridge, MA.
- Kohonen, T. (1995). *Self-Organizing Maps*. Springer Series in Information Sciences.
- Maßmann, A. and S. Posch (1995). Mask-Oriented Groupings in a Contour-Based Approach. In *Second Asian Conference on Computer Vision*, Volume 3, Singapore, pp. 58–61.
- Newell, A. (1990). *Unified Theories of Cognition*. Harvard University Press, Cambridge, MA.
- Rao, R. P. N. and D. H. Ballard (1995). An active vision architecture based on iconic representations. Technical report, University of Rochester, Computer Science.
- Ritter H., Martinetz T., Schulten K. (1990) *Neuronale Netze – eine Einführung in die Neuroinformatik selbstorganisierender Netzwerke*. Addison Wesley Verlag Bonn.
- Ritter H., Kohonen T. (1989) *Self-Organizing Semantic Maps*. Biol. Cybern. 61:241-254.
- Sagerer, G. and H. Niemann (1997). *Semantic Networks for Understanding Scenes*. Plenum, New York.
- Simon, H. (1962). The architecture of complexity. In *Proc., American Philosophical Society, Vol. 26*, pp. 467–482.
- Treisman, A. (1988). Features and objects: The fourteenth bartlett memorial lecture. *Q. J. Exp. Psychol.* 40A(2), 201–237.