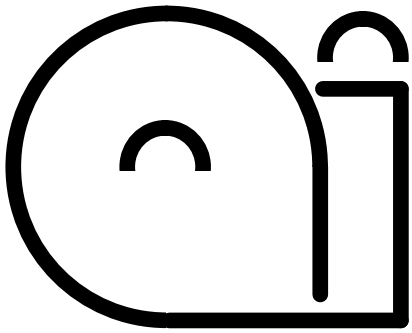



# Österreichisches Forschungsinstitut für / Austrian Research Institute for / Artificial Intelligence



**TR–2005–05**



*Bernhard Jung, Paolo Petta*

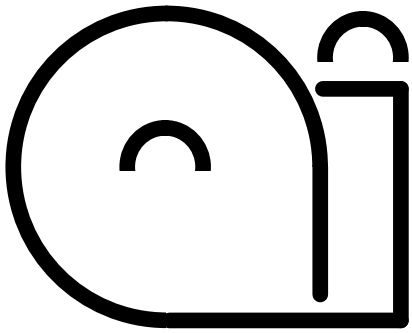

**Agent Encapsulation in a Cognitive Vision MAS**


• Freyung 6/6 • A-1010 Vienna • Austria •

• Phone: +43-1-5336112 •

• mailto:sec@oefai.at •

• http://www.oefai.at/oefai/ •



# Österreichisches Forschungsinstitut für / Austrian Research Institute for / Artificial Intelligence

**TR–2005–05**

*Bernhard Jung, Paolo Petta*

## Agent Encapsulation in a Cognitive Vision MAS

The Austrian Research Institute for Artificial Intelligence is supported by the Federal Ministry of Education, Science and Culture.

*Citation:* Jung B., Petta P.: Agent Encapsulation in a Cognitive Vision MAS. Technical Report, ÃŰsterreichisches Forschungsinstitut fÃijr Artificial Intelligence, Wien, TR-2005-05, 2005, in Pechoucek M., Petta P., Varga L.Z. (eds.): Multi-Agent Systems and Applications IV, 4th International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2005, Budapest, Hungary, September, 2005, Proceedings, Springer-Verlag Berlin Heidelberg New York, pp. 51-61, 2005.

# Agent Encapsulation in a Cognitive Vision MAS

Bernhard Jung[1] and Paolo Petta[1,2]

[1] Austrian Research Institute for Artificial Intelligence (OFAI),
A-1010 Vienna, Freyung 6/6, Austria
[2] Dept. of Med. Cybernetics and AI, Centre for Brain Research, Med. Univ. of Vienna,
A-1010 Vienna, Freyung 6/2, Austria
{bernhard.jung, paolo.petta}@ofai.at

**Abstract.** We cast a baseline cognitive vision design into a multi-agent framework and therein address the questions how and to what extent explicit consideration of coordination may affect the design and performance of such systems. In an analysis of our decomposition into task-dependent entities using both, functional and physical approaches to encapsulation, we show that different kinds of algorithms with different notions of architecture and representation become possible. We describe the evolution of our implementation out of a traditional monolithic design. Functionalities akin to notions of conventional tracking and reasoning now emerge out of the distributed interaction between component agents, with a performance at least on par with the baseline system.

## 1 Introduction

This work was carried out in the context of an Austrian Joint Research Project (JRP) "Cognitive Vision" (see Sect. 6). The domain of Cognitive Vision emerged out of traditional Computer Vision, as "an attempt to achieve more robust, resilient, and adaptable computer vision systems by endowing them with a cognitive faculty: the ability to learn, adapt, weigh alternative solutions, and even the ability to develop new strategies for analysis and interpretation" [ECVision Roadmap V4.2, p.2]. The scientific foundations for Cognitive Vision include visual sensing; architecture; representation; memory; learning; recognition; deliberation & reasoning; planning; communication; and action: issues not independent of each other. Furthermore, the definitions of architecture (a "minimal set of information processing modules and their network of inter-relationships") and representation ("any stable state of a cognitive systems") [ibid. p.11] reflect a classical view. In our interaction with project partners from traditional Computer Vision, we challenge the view that a vision architecture can be seen as a functionally (in the sense of processing or transformation functions) reduced set of processing modules when the system should be goal-directed and purposive; that representations are stable when visual input is noisy, imprecise, or ambiguous; and that vision algorithms necessarily deliver valid results at all times (i.e., are perfect functions).

We show how a traditional vision architecture was recast into a multi-agent design that does not follow the typical functional decomposition into detector, tracker, and reasoner, but employs a combination of the functional and physical approaches to encapsulation [Shen & Norrie 1999, Parunak et al. 2001]. Consequently, within this new

M. Pěchouček, P. Petta, and L.Z. Varga (Eds.): CEEMAS 2005, LNAI 3690, pp. 51–61, 2005.

solution, representation is not a stable state of the system and cannot be pinned down to specific data in some components, but is distributed among agents and their interaction patterns. Following an agent-based design process, we switched from a design perspective that regards vision as a transformation process to one in which cognitive vision is realised as interaction of task- and purpose-dependent agents.

Section 2 describes our point of departure in terms of the scenario and the classical vision solution used to derive and evaluate our approach. In Sect. 3 our MAS-based solution is presented in detail; Sect. 4 reports results; Sect. 5 discusses related work on agent-based systems in computer vision; and Sect. 6 concludes with a discussion and future perspectives.

## 2 The Point of Departure

The starting point of our work was an implementation of the classical functional transformation approach to solve the Hide&Seek M6 (aka "shell game") scenario of the Austrian JRP "Cognitive Vision". It consists of a stationary camera, two black cups and an orange ball on a table, and a single human hand moving and lifting the cups in turn to hide and unhide the ball (see Fig. 1).[1] Questions to be answered by the system include: "Where is the ball?"; "What is hiding the ball?"; and "What was the trajectory of a cup, the ball, or the hand?". To this end, beyond detection and tracking of objects, capability to reason about occlusion is also required.

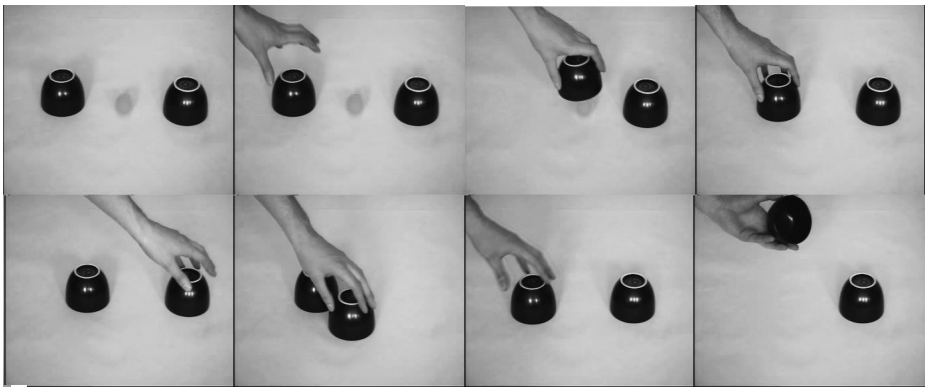


**Fig. 1.** A sequence from the JRP Hide&Seek M6 scenario (covering only part of the activity)

A MatLab-based implementation follows the transformational approach to derive trajectories and occlusion relations of objects in the scene.[2] It comprises three object detectors that perform colour-based segmentation and calculate different properties for the blobs (=contiguous regions) derived, including area; centre of gravity; bounding

[1] To be available at <URL:http://www.acin.tuwien.ac.at/groups/robtec/fsp/fsp.htm>

[2] Provided by A. Opelt, G. Schweighofer, A. Pinz & R. Tomasi from the Institute of Electrical Measurement & Measurement Signal Processing (EMT), Graz University of Technology.

box; and solidity. *Object detection* is then based upon these properties. These detectors are used by *trackers* that consider objects detected in subsequent frames to be the same as long as the distance between their centres of gravity remains below a given threshold. The trackers can thus detect *new* objects as well as the *loss* of previously detected and tracked ones. This information is used by a reasoner to maintain an *occlusion tree* registering which object is currently hidden by which other one. For each lost object, a hiding object is looked for: if successful, the lost object is assigned to its hider and hereafter assumed to move together with it. New objects are first assumed to have been *unhidden* and a match is attempted to hidden object entries in the occlusion tree. If a match is found, the hitherto hidden object is unlinked from its hider and updated with the information about the newly detected object. The new object is thus *unified* with the old one. In this manner, stable reasoning about occlusion of the ball and temporary occlusions by the hand hiding the cups is achieved.

To obtain an efficient baseline system[3], this solution (referred to as "EMT solution") was cast into the JRP's common framework by modularising the monolithic MatLab-based architecture into specialised detectors and associated trackers for balls, cups, and skin; and an occlusion tree reasoner. This application was used subsequently to assess and compare correctness, performance, and architectural features of our new design.

## 3 Tracking and Reasoning by Agent Coordination

MAS *coordination* [Lesser 1998] aims at ensuring coherent behaviour of a system consisting of multiple autonomous agents pursuing interdependent activities—e.g. intending to work on the same or overlapping subproblems; disposing of alternative methods or data to generate a solution; or producing results of one subproblem that also contributes to the solution of another. Coordination typically requires the *detection* of interdependencies; a *decision* which coordination action to apply; and *coordination mechanisms* (for an overview of available techniques, see e.g. [Omicini & Ossowski 2003]) that shape the way the agents perform these tasks. In our architecture, each task-relevant object in the scene is represented by a dedicated agent. These *object agents* are supported by a limited number of specialised *detector agents*, that provide an anchoring to entities detected in the current image. In this design, tracking and reasoning functionalities *emerge* out of the interactions among object agents, and between object and detector agents (see Fig. 2). Roles with responsibilities and authorities are assigned to agent types at design-time (see Sect. 3.1 and 3.2). This leads to organisational restrictions: e.g. each object agent is associated to a specific detector agent. At run-time, agent coordination is guided by creation and termination of agents; auctioning; contracting; and matchmaking. This is implemented via agent communication following a predefined conversation policy [Greaves et al. 1999].

Even though the design of this conversation policy was guided by the ideas of tracking and (occlusion tree) reasoning, and it does include specific "tracking" and "reasoning" phases (see Sect. 3.3), the overall tracking and reasoning capabilities expressed by the system cannot be pinned down to particular component agents, but come

[3] The MatLab code was found to be an order of magnitude slower than its C/C++ equivalent.

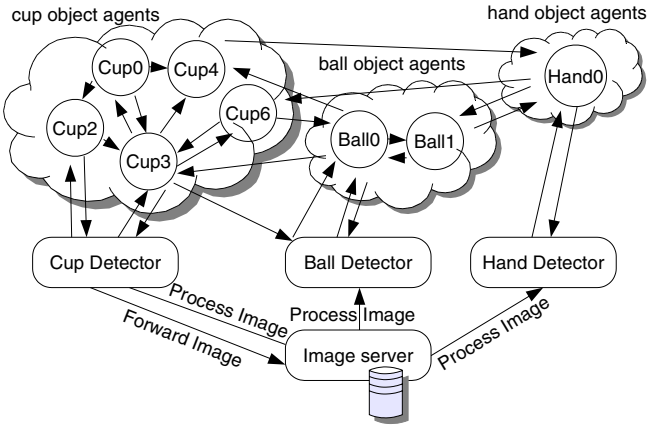



**Fig. 2.** The architecture, consisting of an image server agent and detector agents interacting with object agents (Cup1 and Cup5 are supposed to have come and gone in the observed scene). Unlabelled arrows indicate communication paths.

about in a distributed fashion, relying also on properties (e.g. continuity) of the environment. In the following, we first describe the responsibilities of detector and object agents and then discuss the conversation policy governing agent interaction in more detail.

### 3.1 Detector Agents

A *detector agent* is responsible for the detection of object blobs of a given type in the images provided by an image server agent (connected to a live camera or accessing an image sequence store). It distributes the information about these blobs to interested object agents of the same type (see Sect. 3.2). The interest of an object agent is expressed in terms of multiple regions of interest in the image and is specified on demand.[4] A detector agent is further responsible for mediating among multiple object agents claiming a blob, by holding an *auction* and awarding the blob to the object agent that submitted the bid expressing the highest confidence in this blob representing itself (=the particular object) in the current image (see Sect. 3.3). A detector agent will spawn *new object agents* for unmatched blobs (i.e., not claimed by the existing object agent population), and serves as a *matchmaker agent* [Wong & Sycara 2000] distributing requests to object agents to identify hider objects for objects that disappeared with the current frame. Furthermore, it is the joint responsibility of the set of all detector agent instances to coordinate the tracking and reasoning phases in the conversation policy (see Sect. 3.3).

### 3.2 Object Agents

An *object agent* represents an object detected in the scene and is associated to the detector agent that spawned it. Object agents are responsible for the matching of blobs

[4] The specification of more than one area is required to handle reappearance after occlusion correctly, as explained in the following.

offered by their detector agents to the data maintained locally about the most recent blob of their scene objects and thereby for coherent tracking of objects. The distance between centres of gravity of blobs serves as coherence measure and must lie below a given threshold. This presupposes a certain coherence of information across subsequent frames (cf. end of introduction to Sect. 3). They further handle disappearance and reappearance events in the scene by linking and unlinking themselves to hider objects. Once linked, a *contract* is established [Jennings 1996] between the agents representing the hider and the hidden objects, and the "hider object" agent subsequently propagates position changes to the "hidden object" agents. Object agents also send updates of their areas of interest to their detector agent. These areas are usually extended bounding boxes around currently assumed object positions (and, in the case of hidden objects, also the locations of their disappearance).

By these means, detector agents automatically offer blobs also to object agents representing hidden objects; reappearance can be detected, and no "object merging"—unification of newly instantiated re-appearances of objects with representations of their earlier occurrences—as in the original EMT approach is necessary. If the object referred to by an object agent goes undetected for some time without identification of an appropriate hider, the object agent will eventually assume its object disappeared from the scene for good, and die.

### 3.3 Conversation Policy

Coordination among agent instances is governed by an encompassing conversation policy, articulated into a tracking and a reasoning phase. A simplified diagram of the policy is shown in Fig. 3. The *tracking phase* of the policy is started by the image server agent that sends `ProcessImage` messages to the (three) detector agents available. In case a detector agent fails to identify any blobs of its kind in the current image within the regions of interest of its object agents, it sends them `NoBlobFound` messages, to be confirmed by a `BlobConfirmed` return message. Otherwise, the detector agents try to assign each of the blobs detected to one of their existing object agents, based on the location of the blob and the areas of interest of the object agents being managed: a blob may be offered to multiple object agents, and an object agent may have multiple blobs offered by its detector agent. To this end, detector agents send `DetectedBlobs` messages to their object agents. Each of these returns a `BlobSelection` message, with the index of the blob assessed the most likely reference to the object it represents and a confidence measure[5], or an index value of -1 to report that no blob of interest was identified. The detector agent waits for all `BlobSelection` messages to be returned and resolves ambiguous selections by sending a `ConfirmBlob` message to the object agent that expressed the highest confidence. In subsequent iterations, blobs remaining are offered to object agents not yet awarded a blob, and the sub-policy ends with a (possibly empty) remainder set of not assignable blobs for which the detector agent spawn new object agents. The *tracking phase* thus ends with receipt of the messages confirming

[5] Calculated over the centre of gravity of the blob offered and the last known object position or location of disappearance.

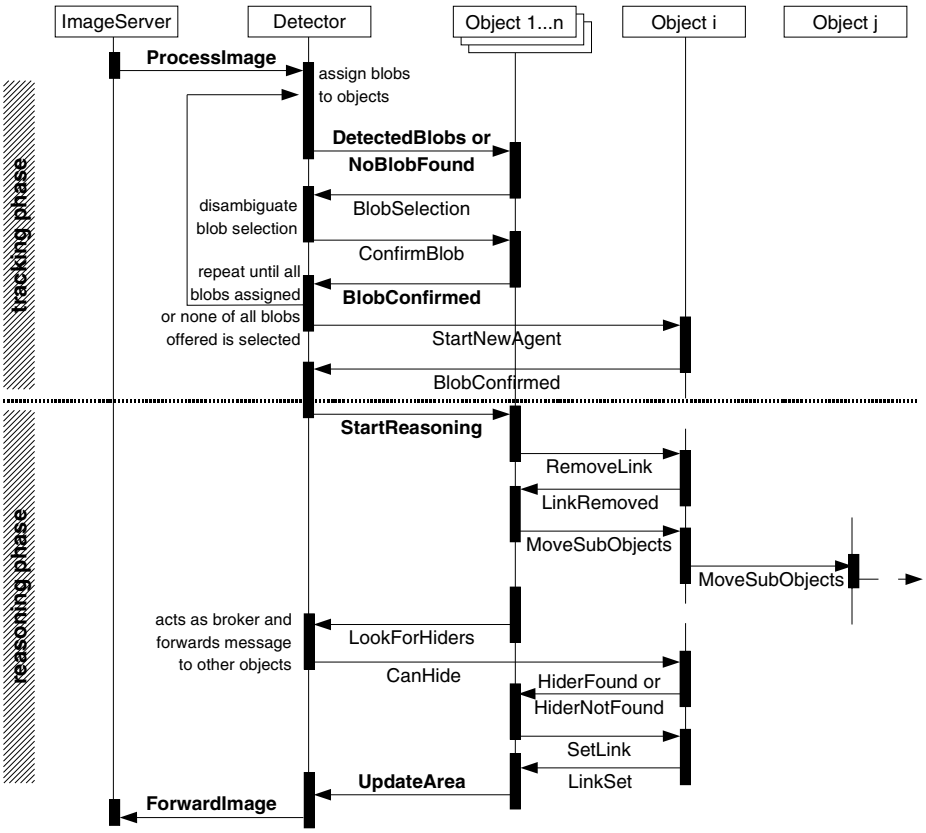



**Fig. 3.** Schematic sequence diagram for the conversation policy. Mandatory messages are set in bold font. Parallel messages, conditions for messages, and synchronisation between detector agents, are omitted for clarity. See main text (Sect. 3.3) for details.

creation of new agents (if any). Before starting the reasoning phase, *all* detector agents must have finished the tracking phase.

In the *reasoning phase*, detector agents act as matchmakers. We adopted this as approach of choice, given that detector agents know about all object agents they have spawned and message broadcasts are not necessarily desirable, as detector agents have information about the area an object is located in and can thus perform a kind of targeted multi-casting for `CanHide` messages (see below). This phase is initiated by the detector agents sending `StartReasoning` messages to all known object agents. These determine whether they have reappeared, in which case they send `RemoveLink` requests to their hider object agents, which confirm link removal (`LinkRemoved`). Next, every hider object agent remaining sends a `MoveSubObjects` message to the hidden object agents linked to it, thus propagating the relative movement of its own object. This message is forwarded recursively to further hidden objects. Acknowledge messages for `MoveSubObjects` are not sent because no explicit synchronisation before further policy steps is required, and we also assume reliable communication.

**Table 1.** Timing results of the baseline and the MAS-based systems

| Framework components | time (s) |
|---|---|
| None (image server only) | 6.0 |
| Baseline detector | 37.5 |
| Baseline detector+tracker | 41.7 |
| All three baseline components | 46.6 |
| MAS-based solution | 40.5 |

| approximate tracking & reasoning times (s) | |
|---|---|
| Baseline solution | 9.1 |
| MAS-based solution | 3.0 |

The next policy step is to identify hider objects for all disappeared objects. Disappeared object agents broadcast `LookForHiders` messages to *all* detector agents; these forward these requests as `CanHide` messages to all objects they know to be in the neighbourhood of the disappeared objects. Potential hiders determine confidence values for them being the cause of specific occlusions and accordingly return `HiderFound` or `HiderNotFound` messages to the disappeared object agent. In the case of multiple possible hiders, the disappeared object agent selects the one with the highest confidence and sends a `SetLink` request to it. This request is confirmed by a `LinkSet` message from the hider object agent. Object agents end their participation in the policy by adjusting their area of interest via an `UpdateArea` message to their detector agent. Upon receipt of all `UpdateArea` messages from their object agents by the different detector agents, the reasoning phase ends, and the image server is asked by means of a `ForwardImage` message to proceed to the next image.

## 4 Results

Performance measurements were obtained for the baseline system (Sect. 2) and our MAS solution (Sect. 3) on a Pentium M 1.6 GHz notebook under Gentoo Linux.[6] Both cases processed 100 frames of the scenario (with all objects visible and one occlusion of the ball); the time of the whole process was measured. To assess relative processing times of the three sub-components in the baseline solution, we added one component after the other. The results summarised in Table 1 clearly identify the detector algorithms as the major bottlenecks. As the same detector algorithms are used in all configurations, the difference of about six seconds between the baseline and the MAS solutions can be attributed to tracking and reasoning. MatLab-generated code seems to be inherently slower than hand-written C/C++ code, so these results cannot be interpreted as the agentified solution being three times faster at tracking and reasoning. However, it does indicate the agentified solution to be no worse. Correctness of both approaches was assessed merely by visual inspection of bounding boxes and names of tracked objects with links to bounding boxes and names of hidden objects overlayed on the image sequence. The approaches show slight differences in the times of object reappearance during phases of occlusion, but all objects are tracked and labelled correctly at the end of the occlusion episode. Concerning architectural features, our approach is easier to

[6] Working memory requirements appear to be secondary, on the order of 40MB resident and 160MB virtual size of a multi-threaded process.

maintain due to the modularisation and decoupling of processing steps (a prerequisite for further work on asynchronous operation in decelerated real-time, Sect. 6), and to extend by adding new detector agents and more complex object interaction models. Scalability can be achieved by distribution of agents over multiple machines.

## 5 Related Work

The modularisation of the EMT solution was guided by the structure of the available MatLab code, resulting in detector, tracker, and reasoner components. Out of these, our final solution uses detector and object agents only. The modules of neither solution correspond directly to what [Boissier & Demazeau 1992] termed *basic agents*, i.e., cells of a two-dimensional matrix of focus (contours, highlights, range data, stereo-vision, ...) and representation (image, image features, scene features, ...) dimensions, following the traditional decomposition of [Marr 1982]. [Bianchi & Rillo 1996] follow up on the work of Boissier & Demazeau and present a distributed control architecture applied to purposive computer vision tasks. The system is specified in terms of a set of behaviours which are decomposed into tasks and delegated by autonomous agents to basic agents. In contrast, our agents follow more or less a functional decomposition (EMT, Sect. 2) or represent task- and purpose-dependent entities (object-agents, Sect. 3). [Graf & Knoll 2000] propose a MAS architecture with a greater degree of flexibility than Boissier & Demazeau and Bianchi & Rillo. Their agents accomplish specific vision tasks by a goal-driven communication process. Master agents with complex planning and interpretation capabilities are distinguished from slave agents encapsulating image processing algorithms. The papers of a special issue of Pattern Recognition on agent-based computer vision [Rosin & Rana 2004] provide a recent snapshot of the state of the art in the area. They demonstrate how MAS technology can be applied to a variety of vision tasks while underscoring that the full potential of the agent and multi-agent paradigm is still to be uncovered. The editorial addresses also the criticism of agent-based systems that *"they are just an elaborate and unnecessary metaphor, and often do not actually provide better results than traditional techniques"* [ibid.], suggesting that agent-based systems should be considered as an alternative approach to computer vision, and that *"it would be useful for the vision community to consider the full potential of the "agent" and multi-agent paradigm"* [ibid.] The history of MAS in vision reflects to some extent the evolution from strict control to increasingly flexible coordination; from decomposition according to [Marr 1982] (transformation functions) to increasingly more flexible run-time behaviours (online service realisation). In our approach, we try to go a step further and provide a different perspective by moving from functional, transformational decompositions that can be aggregated to service realisation to task- and purpose-dependent agents (cf. [Shen & Norrie 1999, Parunak et al. 2001]).

Relating the idea of representation as a stable state of a cognitive system as defined in [ECVision Roadmap V4.2, p. 11] to our approach reveals that representation cannot be pinned down to particular data in specific agents. The concept of occlusion for instance is "represented" in the contracts between the hider and the hidden objects: it

evolves dynamically over time. We cannot look at the system's state at a certain time and derive occlusion relations from it. A contract might be setup by momentary failure of a detector leading to the illusion of an occlusion; this fact is nothing but a distraction when the task is to interpret the scene, and this bad contract is soon resolved as the object is detected again. An occlusion relation requires some stability of information in the agents and interaction patterns between them to arise. It cannot be found in the micro-states of an agent or the states of a conversation, nor is it modelled explicitly as a reified macro-state. Even so, this dynamic and active form of representation allows for the system's interpretation of occlusion.

Due to our research focus on architectures for building computer vision systems rather than modelling cognitive behaviour, our approach is not directly comparable to typical cognitive architectures like ACT-R or SOAR. At this early stage, we also do not yet have entities comparable to short/long-term memory or explicit knowledge. Decentralisation and decoupling in our solution may remind of distributed blackboard based interpretations [Lesser & Erman 1980] or blackboard architectures in general (e.g. [Hayes-Roth 1995]). There are important differences between HEARSAY and our approach concerning both the domain and the processing: Relations between utterances occur over time only; in the visual domain relations occur over both time and space and are mediated via two-dimensional image sequences. Hypothesis processing in HEARSAY is transformational, based on grammar hypotheses, and not interpretational; while HEARSAY focuses on constraining the search space of a given problem, we focus on permanent (re-)interpretation of continuous input, that is driven not by internal state but by events in the scene.

## 6 Discussion and Outlook

We have shown that taking the agent perspective on computer vision can lead to algorithms with different decompositions (reflected in the scope and responsibilities of agents) and to solutions that are inherently distributed. Even so, the solution presented is still a far cry from exploiting the full potential of MAS. In the following, we discuss selected aspects, some limitations, and conclude with an outlook of how to try overcome them.

The conversation policy of our current design has two major synchronisation points[7], one after the tracking phase, and the other after the reasoning phase. Both are necessary to perform consistent tracking and reasoning and to keep in sync with the image data. Whether and how this explicit synchronisation can be removed—e.g., by another conversation policy or intelligent scheduling—is subject of future work. Implementation of conversation policies can be facilitated by framework support for hierarchical finite state machines. As this support is currently not available in our framework, the actual implementation is not well structured, and the description of the conversation policy was assembled from existing code and design fragments that guided the implementation. The autonomy of agents in our system is not yet an absolute necessity for the kind of coordination we investigate, as various timing issues have been ignored for the

[7] There are also some minor ones, e.g. after link removal for a disappeared object, right before the `MoveSubObject` messages.

time being. In particular, all agent behaviour is synchronised with the video frame rate and not real-time. Nevertheless, the agent paradigm already is of value in the design of the overall system and of the interactions to be coordinated.

As an alternative to the agentified solution, one could come up with two functions corresponding to the tracking and reasoning phases that perform the same kind of tracking and reasoning our system does. Arguably, these could be easier to implement, not requiring an agent framework nor mechanisms for explicit synchronisation with the image stream. We propose, however, that a main feature of our approach lies in the change of viewpoint on the problem of tracking and reasoning it affords. Although the design of appropriate conversation policies requires more (or at least different) skills and leads to a complex system, it does simplify the integration of heterogeneous detectors and coverage of specific objects (e.g., objects not capable of hiding or being hidden). Even so, we grant that thinking and designing locally from an agent's viewpoint as well as locally in space (in terms of regions of interest), requires non-local dependencies to be captured explicitly (e.g., the impact of changes in lighting or cast shadows) and may lead to unpredictable (for better or worse) system behaviour (a typical problem of complex systems). But while these properties of such a MAS may be seen as disadvantages when building industrial systems, they may in fact be an important advantage when it comes to grasping the complexity of vision. As the current results show, the presented agentified solution does not differ significantly from the original design in terms of correctness or speed while introducing a new perspective on decomposition of vision systems.

## Acknowledgements

The authors would like to acknowledge the discussions and interactions within the Cognitive Vision JRP, that all formed valuable contributions to this work. The original EMT solution was provided by A.Pinz (FWF S9103-N04) and R.Tomasi. JRP partner ACIN provided zwork as the common framework of the JRP. This research was carried out in the context of the projects S9103-N04, S9106-N04 and S9107-N04 of the FWF Austrian Science Fund. The Austrian Research Institute for Artificial Intelligence is supported by the Austrian Federal Ministry for Education, Science and Culture and by the Austrian Ministry for Transport, Innovation and Technology.

## References

[Bianchi & Rillo 1996] Bianchi R.A.C., Rillo A.H.R.C.: A Distributed Control Architecture for a Purposive Computer Vision System, *Proc. IEEE Intl. Joint Symposia on Intelligence and Systems (IJSIS'96)*, 288–294, 1996.

[Boissier & Demazeau 1992] Boissier O., Demazeau Y.: A Distributed Artificial Intelligence View on General Purpose Vision Systems, in Werner E., Demazeau Y., Decentralized AI 3, North-Holland Amsterdam/New York, 311–330, 1992.

[ECVision Roadmap V4.2] A Research Roadmap of Cognitive Vision. ECVision: The European Research Network for Cognitive Computer Vision Systems. IST–2001–35454. 4.2 11–2–05, 2005. <URL:http://www.ecvision.org/research_planning/ECVisionRoadmapv4.2.pdf>

[Graf & Knoll 2000] Graf T., Knoll A.: A Multi-Agent System Architecture for Distributed Computer Vision, *International Journal of Artificial Intelligence Tools*, 9(2):305–319, 2000.

[Greaves et al. 1999] Greaves M., Holback H., Bradshaw J.: What Is a Conversation Policy?, in Bradshaw J. et al. (eds.), Workshop Notes, "Specifying and Implementing Conversation Policies", Third Intl. Conf. on Autonomous Agents (Agents '99), Seattle WA, 1999.

[Hayes-Roth 1995] Hayes-Roth B.: An Architecture for Adaptive Intelligent Systems. Artificial Intelligence **72**:329–365, 1995.

[Jennings 1996] Jennings N.R.: Coordination Techniques for Distributed Artificial Intelligence, in O'Hare G.M.P., Jennings N.R. (eds.), Foundations of Distributed Artificial Intelligence, Wiley Chichester/London/New York, 187–210, 1996.

[Lesser 1998] Lesser V.: Reflections on the Nature of Multi-Agent Coordination and Its Implications for an Agent Architecture. *AAMAS*, **1**(1):89–111, Kluwer Academic Publishers, 1998.

[Lesser & Erman 1980] Lesser V., Erman L.D.: Distributed Interpretation: A Model and Experiment, *IEEE Transactions on Computers*, **29**(12):1144–1163, 1980.

[Marr 1982] Marr D.: Vision, Freeman and Company, New York, 1982.

[Omicini & Ossowski 2003] Omicini A., Ossowski S.: Objective versus Subjective Coordination in the Engineering of Agent Systems, in Klusch M. et al.(eds.): *Intelligent Information Agents: The AgentLink Perspective*. LNAI 2586. Springer Berlin Heidelberg, 179–202, 2003.

[Parunak et al. 2001] Parunak H.Van Dyke, Baker A.D., Clark S.J.: The AARIA Agent Architecture: From Manufacturing Requirements to Agent-Based System Design. *Integrated Computer-Aided Engineering.* **8**(1):45–58, IOS Press Amsterdam, 2001.

[Rosin & Rana 2004] Rosin P.L., Rana O.F. (eds.): *Pattern Recognition*, Special Issue on Agent Based Computer Vision, **37**(4):627–855, 2004.

[Shen & Norrie 1999] Shen W., Norrie D.H.: Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey, *Knowledge and Information Systems*, **1**(2):129–156, Springer Berlin Heidelberg, 1999.

[Wong & Sycara 2000] Wong H., Sycara K.: A Taxonomy of Middle-Agents for the Internet, in Durfee E. et al. (eds.), Proc. 4th Intl. Conf. on MultiAgent Systems (ICMAS-2000), July 10–12, 2000, Boston, MA, IEEE Press, New York, NY, 465–466, 2000.