Integration and Coordination in a Cognitive Vision System

Sebastian Wrede, Marc Hanheide, Sven Wachsmuth and Gerhard Sagerer
Applied Computer Science, Bielefeld University, Faculty of Technology,
P.O. Box 100131, 33501 Bielefeld, Germany
{swrede,mhanheid,swachsmu,sagerer}@techfak.uni-bielefeld.de

Abstract

In this paper, we present a case study that exemplifies general ideas of system integration and coordination. The application field of assistant technology provides an ideal test bed for complex computer vision systems including real-time components, human-computer interaction, dynamic 3-d environments, and information retrieval aspects. In our scenario the user is wearing an augmented reality device that supports her/him in everyday tasks by presenting information that is triggered by perceptual and contextual cues. The system integrates a wide variety of visual functions like localization, object tracking and recognition, action recognition, interactive object learning, etc. We show how different kinds of system behavior are realized using the Active Memory Infrastructure *that provides the technical basis for distributed computation and a data- and event-driven integration approach.*

1 Introduction

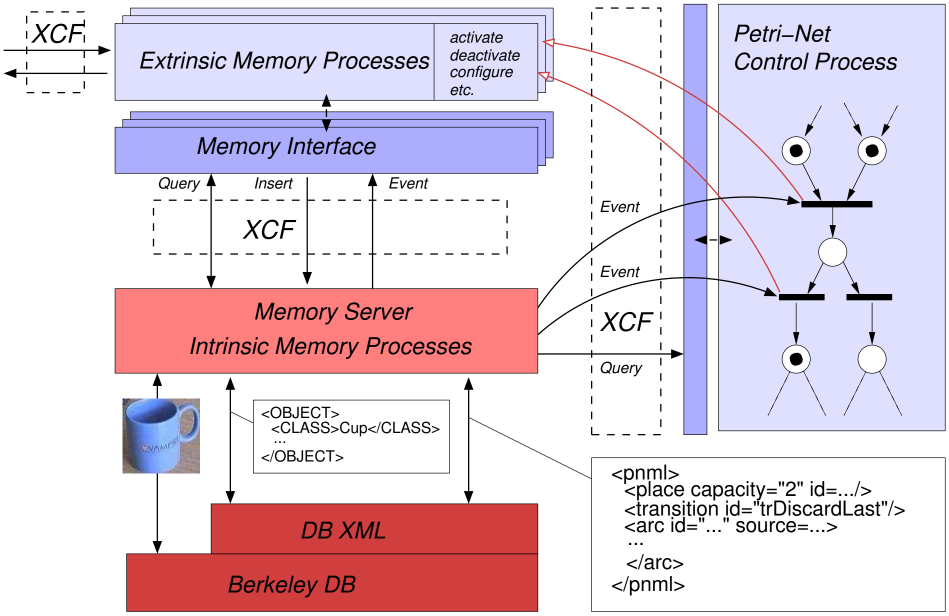
Pushing the construction of computer vision systems from heuristically motivated approaches to a systematic engineering discipline has been a goal of computer vision research for quite a long time. Early approaches merily focused on the knowledge engineering task using general inference engines [6, 8, 12, 24]. These had their own limitations when going to broader applications, more unrestricted realistic environments, and real-time constraints. In such broader settings, knowledge can not be defined sufficiently neat and crisp. Furthermore, in many cases it revealed to be difficult to separate system control from domain knowledge. As a consequence the interest shifted from generic computer vision systems towards specialized techniques and representations for individual object recognition [20, 26, 19] and solving more specific vision tasks [1], e.g. vehicle guidance [15], people tracking [25], etc. Furthermore, the paradigm of understanding computer vision as an active or interactive process poses real-time and hardware requirements to at least components of a vision system. Being able to solve more specific tasks in more unrestricted settings, the general idea is to generate more complex and more general application systems by combining a wide variety of different specialized vision behaviors.

Following a system approach in computer vision research, the topics of distributed processing, communication between components, integration, and coordination are becoming major issues in the design of computer vision systems. How to relate these basic technologies to the solution of computer vision problems has been rarely studied (different aspects can already be found e.g. in [4, 11, 14]), but it gains much interest in the emerging field of cognitive systems [10, 29, 31, 22]. However, use cases of building complex vision systems that are actually running in a real-world scenario and integrate a large number of different vision behaviors in a unified framework are only sparsely reported. A few examples that present complex integrated systems can be found in [18, 4, 2].

In this paper, we present an AR system that realizes a cognitive assistant for mixing drinks that includes low-level as well as high-level visual processing. The system integrates real-time components, human-computer interaction loops, the processing of dynamic 3-d environments, and information retrieval capabilities. Processing modules are decoupled through a repository-style architecture and distributed over several computing machines. System integration and coordination is managed by using XML as a unified data-model and declaratively defined memory events. These provide the technical basis for a petri-net component that manages task-dependent control issues.

In the following sections, we sketch the integration and coordination approach used in system construction and describe the demonstration system in detail. Exemplarily, three different system behaviors and corresponding processing paths are discussed that illustrate the integration principles.

0-7695-2506-7/06 $20.00 © 2006 IEEE



```
<OBJECT>
  <CLASS>Cup</CLASS>
  ...
</OBJECT>
```

```
<pnml>
  <place capacity="2" id=.../>
  <transition id="trDiscardLast"/>
  <arc id="..." source=...>
  ...
  </arc>
</pnml>
```

Figure 1. Integration components of the Active Memory Infrastructure

2 The Active Memory Concept for System Integration

The interactive application scenario of our cognitive assistant system described in section 3 in particular, and interactive cognitive vision systems in general, impose several constraints on a software framework for system integration. For us, two of the most important requirements are firstly that the system should interact with soft real-time performance and secondly the ability of the system to track the interaction context in terms of perceived episodes, events and scenes.

2.1 Data-driven integration

Given these requirements, we developed an integration framework for cognitive vision systems [30], called *Active Memory Infrastructure*[1] (AMI). The basic concepts followed there are the ability to integrate distributed processes with various communication patterns useful for vision systems and an active data repository allowing for flexible knowledge representation. To achieve this needed flexibility all information flow (e.g. object recognition results) between integrated components is based on XML messages that can reference attached binary data (e.g. images).

Utilizing these XOP-like data packages [27], the *XML enabled Communication Framework* (XCF) [29] supports (a-)synchronous remote method invocation (RMI) and publisher-subscriber communication semantics to distribute components over several computing nodes. A component interface provides AMI processes with default implementations for external process control and reconfiguration. Exposed methods are bound and invoked dynamically, with XML schemas optionally providing runtime type safety of exchanged parameters.

On top of the communication framework, the *Active Memory XML Server* serves as the basis for coordination and shared data management in our integration approach as shown in figure 1. XML data like object or action recognition results and/or binary data like image patches for recognition algorithms are fed into an active memory server and can be retrieved via XPath statements.

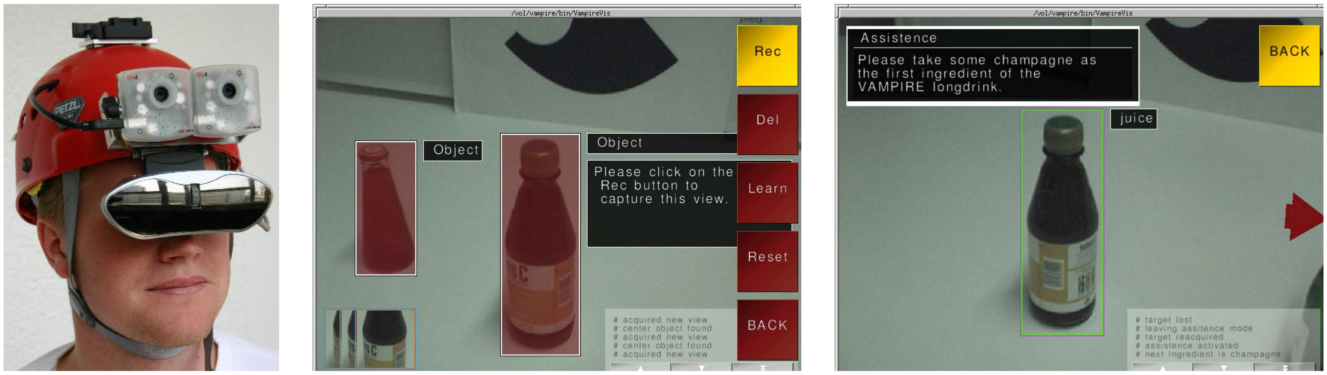
2.2 Event-driven coordination

Basic coordination between the components is provided by a flexible event-notification mechanism. The event manager of the active memory server is co-located with the persistent back-end, a native XML database. Event subscriptions specify an XPath to narrow down documents of interest and the associated type of database action. If an action in the active memory server matches an event subscription, the subscriber is notified, if available. Coordination is thus data-driven and *not* bound to explicit links between a fixed set of components present in the system.

To provide more complex coordination methods for multiple concurrent components in a cognitive vision system, we developed an application of petri-nets for system integration purposes. Petri-nets, in general [21], extend clas-

[1] The resulting SDK is available for download at: `http://xcf.sourceforge.net`

0-7695-2506-7/06 $20.00 © 2006 IEEE



Please click on the
Rec button to
capture this view.

```
# acquired new view
# center object found
# acquired new view
# center object found
# acquired new view
```

Please take some champagne as
the first ingredient of the
VAMPIRE longdrink.

```
# target lost
# leaving assitence mode
# target reacquired
# assistence activated
# next ingredient is champagne
```

(a) AR gear

(b) Training new objects interactively

(c) The system guides the user to the next ingredient

Figure 2. The assistance system: Hardware setup and screenshots.

sic state machines by the ability to represent concurrency. Thus, they are well suited for modeling structure and behavior of parallel distributed systems. The current marking of a petri-net corresponds to a specific system configuration. Dynamic changes in system behavior are controlled by activated transitions. In our approach we extended the classical petri-net concept by so-called *active memory guards* (AMG) which utilize memory event listeners to connect the model to active memory instances. If the place condition is fulfilled and the specified memory event occurred, the input arc as a whole is satisfied and the attached transition is enabled. This concept couples the execution of the specified high-level petri-net model to the overall state. AMGs are context dependent in a sense that they are activated as soon as the place condition of its arc is satisfied.

The realization of the active memory petri-net engine allows a formal and declarative specification of net structure and active memory guards as well as the attached actions in an application of the PNML document format [28]. Thus, petri-net coordination models can be extended by new places and transitions online. As soon as a PNML model is updated in an active memory server, the instantiated petri-net execution engine is reconfigured.

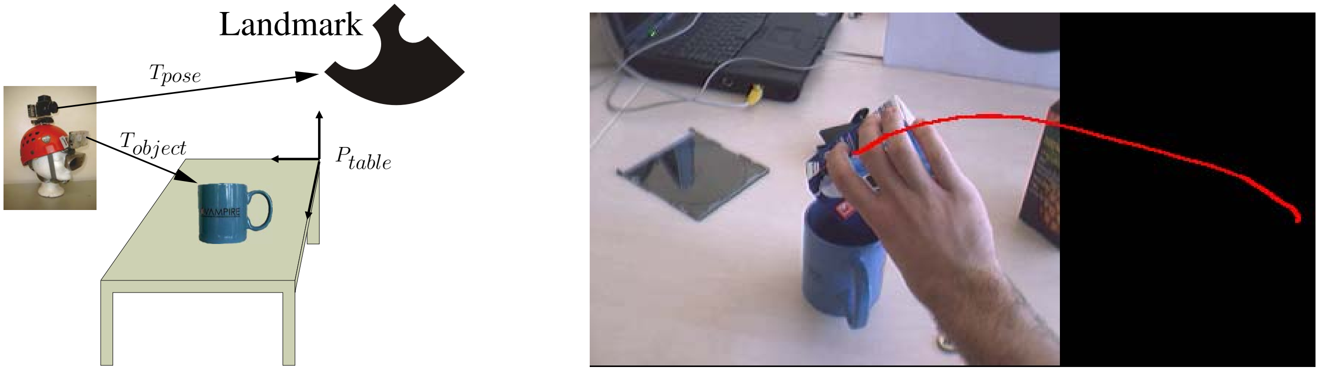
Figure 1 shows on a conceptual level how a petri-net control process is coupled to the overall system. As soon as a transition fires, a sequence of actions is executed. The set of possible actions which can be attached to a transition can be any number of XCF RMI calls, basic actions on an active memory or local calls to methods of classes that are derived from a basic action interface. Instances of those actions are specified in the PNML model and can be configured with XML parameters. A concrete example for modeling of system behavior with petri-nets will be given in section 4.3 along the explanation of our system architecture.

3 The Cognitive Assistant System

An approach for system integration can only be proved appropriately by realizing integrated systems. In this paper we present how the active memory concept is employed to construct a cognitive assistant for mixing drinks. The user wears an augmented reality setup as depicted in figure 2(a). Images captured by the two front-mounted cameras are augmented by additional information and displayed on the head-mounted display of the so-called AR gear. This device thus realizes the interface between the system and the user by means of GUI elements and visual highlights as depicted in figures 2(b) and 2(c). As feedback channels a microphone and a wireless mouse are integrated.

In the scenario, a user A first has to teach the system the ingredients of the drink by interactively capturing about 4-5 views of each object. Therefore, the user is prompted to focus his or her view on the object to be learned. Acquired views are presented in the head-mounted display for validation (cf. figure 2(b)). The acquired image patches are used as training set which is labeled by speech (e.g "This is orange juice."). Afterwards, the system is able to recognize these objects and memorize and update their 3D position autonomously. Another user B can now use the system as an assistant to really prepare a drink. The system prompts the user to follow the recipe step by step and supervizes whether he or she is correctly performing. Furthermore, the system provides assistance in guiding the user visually by means of augmented reality arrows (see right of figure 2(c)) to the memorized location of the next ingredient.

To achieve the described functionality of a cognitive assistant several components have to play together. Each of these components usually reflects an area of research for itself. We will briefly outline the most important components for this scenario in the following.



(a) 3D vision

(b) Trajectory based action recognition

Figure 3. Components of the integrated system

Object Recognition & Learning Objects (ingredients) play a crucial role as the system needs to know where which object is located and which objects are manipulated by the user. The integrated appearance based object recognition subsystem [5] is based on a two-step procedure - segmentation and classification. The first is based on the integration of different saliency measures such as local entropy, symmetry and Harris' edge-corner-detection into an attention map. This allows to segment objects located on a less-textured table top. For the classification of the segmented image patches a combination of Vector Quantization, Local Principal Component Analysis, and Local Linear Maps is utilized. These classifiers can be learned interactively with only few views. The training set is automatically extended by including rotated and scaled versions of the captured image patches. The classification itself performs at real-time on recent computers, allowing to apply this approach in the online reactive system.

3D Vision Sub-System The system needs to know the position of objects in the real world to guide the user. Since the environment is perceived only from (visual) sensors mounted to the AR gear its position with respect to the environment must be known. The 3D pose is computed from artificial landmarks [7] as depicted in figure 3(a). To avoid deficits in visual tracking of the landmark, an inertial tracker located at the top of the AR gear aids the tracking process [23]. By means of this hybrid tracking approach, the precise position and orientation of the user T_{pose} can be computed yielding only a very small relative mean distance error of 0.5%. The availability of the user's pose allows the system to compute the 3D position T_{object} of objects located on the table top (or any other known plane) by intersecting the view ray determined by the object position in the image and the pose with the known table plane P_{table}.

Visualization & Multimodal Interaction The AR gear realizes the interface between user and system. It can guide the user visually to certain places, and provides feedback about the system's status and processing result by means of visualization. Since the system and the user share the same view, the scene is indeed augmented by visual elements like semi-transparent rectangles, three dimensional arrows, etc. (cf. figure 2(b) and 2(c)). Furthermore, the system is able to interact with and receive input from the user. The component is designed for multimodal interaction comprising GUI interaction using a mouse wheel or head gestures [16], and speech input [13]. By means of this it enables interactive learning and labeling of objects, visual user guidance and overall control of the system.

Action Recognition As the system should not only guide the user but also should supervise his actions, a component for action recognition is integrated. It has to answer the question whether the user has performed the requested action with the correct object or not. We utilize a classification approach based on the trajectory of the manipulated object in the video sequence [17]. It is trained with model trajectories of the respective actions and copes with variations of these by classifying them using a condensation algorithm. Since objects cannot be reliably recognized by the object recognition component when being manipulated, visual object tracking [3] is integrated to provide the trajectory of the object as input for the action recognition. Whenever an object is reliably recognized, visual tracking is initialized and tracks the object. The robustness of the used approach against occlusion allows to track the object even when being manipulated. A visual background movement model allows to estimate the absolute trajectory compensating the user's own movement. Figure 3(b) shows an estimated absolute trajectory of an object when performing a "pouring" action.

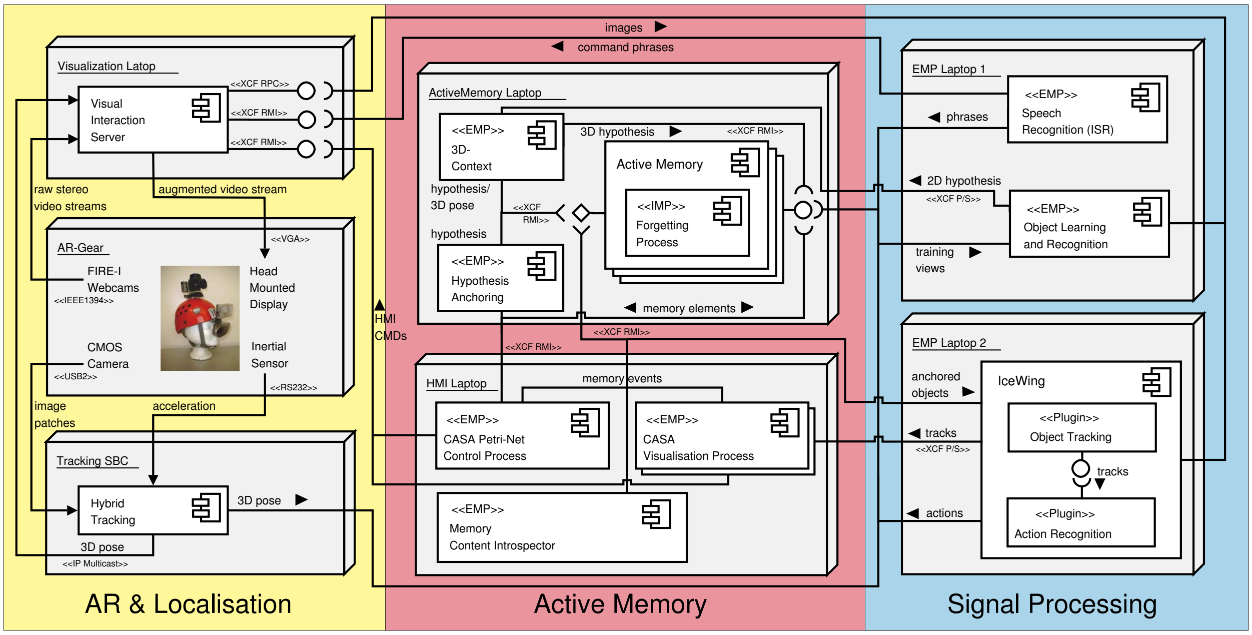


Figure 4. Architectural sketch of the cognitive assistant

Hypothesis Anchoring Components, as for instance object recognition, only provide instant percepts of the environment that describe the current visual appearance of the scene. Inspired by the work of Coradeschi and Saffiotti [9] a component called *hypothesis anchoring* maps these percepts to reliable symbols. For objects, anchoring mainly compares the 3D position of a percept to assign them to existing anchored hypotheses. If no anchored hypothesis matches, a new one is created. Thus, hypotheses are anchored over time and a specific hypothesis gains increased reliability if many matching percepts support it. The reliability factor is included in the hypothesis representations in the active memory. Details about the hypothesis concept and the role of reliability factors in the active memory concept can be found in [30].

4 Architecture and Data-Flow

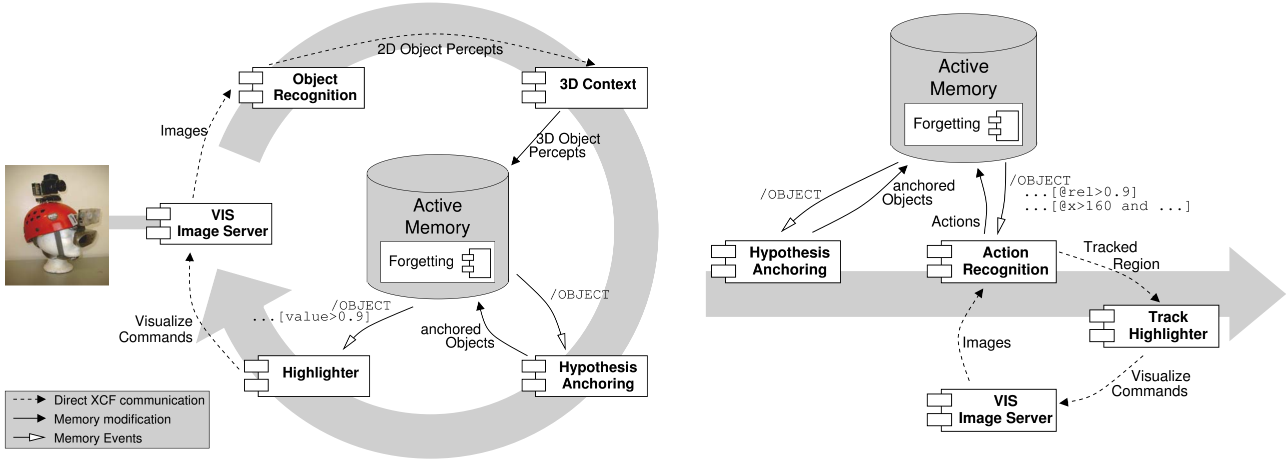
The functionality of the system does not only depend on the individual components, but even more on their adequate and efficient interplay. In the assistance system presented, we utilize the active memory infrastructure for the integration of the different components. Figure 4 presents an architectural sketch of the whole integrated distributed system running on six computers. Describing this architecture at all levels of detail would go beyond the scope of this paper. Rather, three scopes of the system are subject to further explanations about how components work together.

4.1 From images to visualization

Many interesting aspects of how our system mediates data and coordinates components can be explained by following the path of the visual percept of an object from being captured by the camera to its visualization. In figure 5(a) this path is outlined. The component "VIS/Image Server" is connected to the AR gear and serves images as well as accepts visualization commands to display information to the user. By means of this, it closes the interaction cycle. "Object Recognition" recognizes objects in the image and is directly connected to "3D Context" where the 2D percept is extended with 3D information. Here, the active memory (AM) comes into play. The percept is inserted in the AM and because "Hypothesis Anchoring" has subscribed itself on the insertion of such percepts it gets triggered, matches the percept to anchored hypotheses and assigns a reliability to the selected one. The hypothesis is then submitted to the AM again. Thus, "hypothesis anchoring" constitutes a memory process in the sense of the AM concept, as it only works on the memory content.

Following the path further, the hypothesis triggers the "Highlighter" component only if the hypothesis is reliable, since the user should not be bothered with unreliable information. In the AM concept this filtering is realized by registering the component with a more restrictive XPath as shown in the following example: `/OBJECT[RELIABILITY@value >= 0.9]`. Thus, the data is already interpreted by the AM itself. Finally the "Highlighter" calls a remote method on the visualiza-

0-7695-2506-7/06 $20.00 © 2006 IEEE



(a)

(b)

Figure 5. Data mediation: (a) From images to visualization. (b) Triggering components.

tion server "VIS" to display the anchored, reliable object hypothesis to the user. In our integrated system, the complete cycle including object recognition and visualization takes about 0.28 seconds.

4.2 Triggering action recognition

As a second case study we present how action recognition gets triggered in the system. We follow the idea that a user usually focuses on an object before starting to manipulate it. Therefore, the component "Action Recognition" registers itself on reliable (`/OBJECT[RELIABILITY@value >= 0.9]`) and centered (`...[@x>160 and @x<240]...`) hypotheses that are available in the active memory. Figure 5(b) illustrates the complete flow of data in this case study. The "Action Recognition" starts tracking the object in the video stream when it gets triggered by the active memory. The visualization of the tracked regions provides feedback to the user and allows him to follow the system behavior. A recognized action is inserted into the AM.

4.3 Coordinating complex behaviors

Event- and data-driven notification of components as described above is often sufficient for control of individual components. To realize more complex coordination of several components running in parallel, we utilize the petri-net based coordination engine as described in Section 2.2.

To exemplify this, Figure 6 shows a small module of our high-level petri-net that models an exemplary part of the system behavior: The handling of self-localization errors of the 3D vision subsystem. When the user is mixing a drink, the system guides him with arrows to the next ingredient as shown in Figure 2(c). For this task, a correct 3D-pose is necessary. If it gets lost, e.g. due to occlusion of the target, the system has to cope with this situation and reconfigure several system components, e.g. the 3D guide widget in the visualization server. When the pose is again available, the system has to resume normal operation.

Figure 6(a) shows the system working when the pose is available and the 3D object guide is activated. If the AMG of the transition TargetLost is triggered in this state, the 3D context module has inserted information about an illegal pose in the specified memory instance. Thus, the transition fires, which leads to a reconfiguration of the system components and petri-net model state as shown in Figure 6(b). A consequence of this model change is that the transition StopGuide is now fireable. After this transition fires, the guide is paused which is directly reflected in the model as illustrated in Figure 6(c). The system now waits for reacquisition of the 3D pose and in case one is inserted, TargetFound and RestartGuide would be fired and their set of actions be executed. This change would result in the original marking as shown in 6(a).

As described in Section 2, a sequence of actions is executed when a transition fires. To give an example, a parameterized XCF RMI call is attached to the TargetLost transition to deactivate the 3D object guide on the VIS Image Server component. With the set of supplied basic actions most tasks in our system could be carried out and declaratively specified in the PNML model. Custom actions have been added e.g. to control the training process of new objects with the object recognition component.

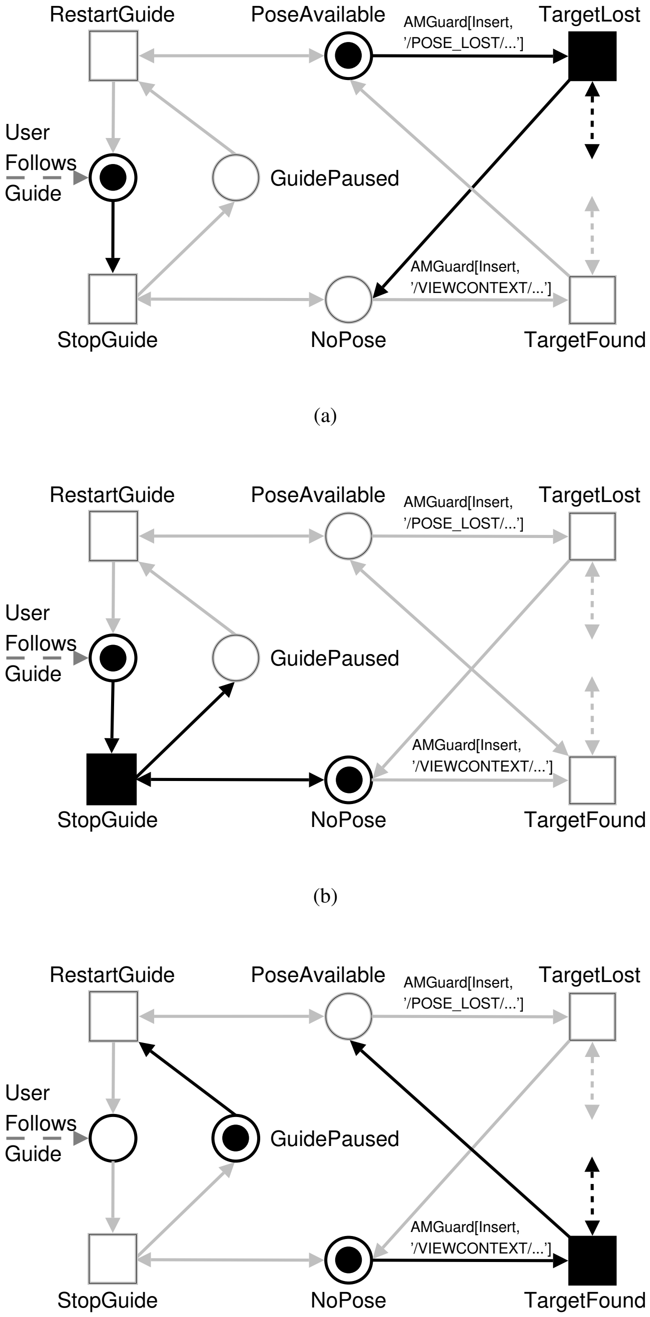


Figure 6. Active petri-net transitions when 3D pose is lost during object guidance. Rectangles depict transitions, circles places and filled circles tokens in places. Relevant model elements of each step are drawn in bold face. Existing AMG specifications are annotated at corresponding input arcs.

5 Experiences and Conclusion

This contribution demonstrated the application of a data- and event-driven integration approach for the development of an interactive vision system for user assistance. We highlighted several typical use cases for system integration in vision systems and their realization with the AMI framework. The resulting system and previous prototypes have already been demonstrated successfully at various occasions, e.g. the EU IST Event 2004 in The Hague, Netherlands as well as on other international research workshops and is planned to be shown at the ICVS 2006 exhibition.

Beyond successful demonstration of the resulting integrated systems being a proof of concept, developer feedback has been very valuable. First of all, the direct use of XML data to encode *information* and not only use it as a data-exchange protocol was reported very useful, especially combined with the features of the active memory and its abilities to process very specific XPath specifications for queries and event listeners. Additionally, extensibility and human readability of the exchanged XML data types directly paid off in shorter development cycles during integration, because of the ability to view messages at runtime. The development process furthermore benefits from the decoupling of components through the active memory. The transport and processing of XML data has been no bottleneck for system reactivity due to the separation of structured and binary data contents.

The application of petri-nets for system coordination allows rigorous modeling and simulation of overall system behavior before a concrete implementation is carried out. The execution engine can call any type of component in a generic manner which means that most component implementations can be directly reused. Furthermore, the concept of active memory guards provides a generic semantic coupling of this model to actions executed in an integrated vision system in order to achieve a specific task behavior.

Future work on the integration framework will focus on an extension of the petri-net engine to colored petri-nets with typed XML tokens for intra-model data exchange yielding more powerful parameterized AMGs as well as more complex basic actions. Regarding the integrated system, user studies are currently carried out on human-computer-interaction patterns which are typical for this new type of proactive vision system.

Finally, we think that our concept enables the shift from purely technical message passing or remote invocation concepts to a more declarative information-driven integration approach for cognitive vision systems. We also expect that the vision of self-adapting and introspective architectures will become feasible along the lines of the presented concepts.

Acknowledgments

The work reported here results from the project *Visual Active Memory Processes and Interactive REtrieval* (VAMPIRE) which has been funded within the European Union IST Program (IST-2001-34401).

References

[1] Y. Aloimonos. Active vision revisited. In *Active Perception*. Lawrence Erlbaum, 1993.

[2] H. Asoh, Y. Motomura, F. Asano, I. Hara, S. Hayamizu, K. Itou, T. Kurita, T. Matsui, N. Vlassis, R. Bunschoten, and B. Kröse. Jijo-2: An office robot that communicates and learns. *IEEE Intelligent Systems*, 16(5):46–55, 2001.

[3] F. Bajramovic, C. Gräßl, and J. Denzler. Efficient combination of histograms for real-time tracking using mean-shift and trust-region optimization. In *Proc. Pattern Recognition Symposium (DAGM)*, Heidelberg, 2005. Springer. to appear.

[4] C. Bauckhage, G. Fink, J. Fritsch, F. Kummert, F. Lömker, G. Sagerer, and S. Wachsmuth. An integrated system for cooperative man-machine interaction. In *Proc. CIRA*, pages 328–333, 2001.

[5] H. Bekel, I. Bax, G. Heidemann, and H. Ritter. Adaptive computer vision: Online learning for object recognition. In *Proc. Pattern Recognition Symposium (DAGM)*, volume 3175 of *LNCS*, pages 447–454. Springer, 2004.

[6] R. A. Brooks. Model-based three dimensional interpretations of two dimensional images. *IEEE Pattern Analysis and Machine Intelligence*, pages 140–150, March 1983.

[7] M. Chandraker, C. Stock, and A. Pinz. Real time camera pose in a room. In *Int. Conf. on Computer Vision Systems*, volume 2626 of *LNCS*, pages 98–110, April 2003.

[8] V. Clément and M. Thonnat. Integration of image processing procedures, OCAPI: a knowledge-based approach. *Computer Vision Graphics and Image Processing: Image Understanding*, 57(2), March 1993.

[9] S. Coradeschi and A. Saffiotti. Perceptual anchoring of symbols for action. In *Proc. Intl. Conf. on Artificial Intelligence*, pages 407–416, 2001.

[10] J. Crowley and P. Reignier. Dynamic composition of process federations for context aware perception of human activity. In *International Conference on Integration of Knowledge Intensive Multi-Agent Systems*, volume 30, pages 300–305, 2003.

[11] B. Draper, J. Bins, and K. Baek. Adore: Adaptive object recognition. *Videre*, 1(4):86–99, 2000.

[12] B. Draper, R. Collins, J. Brolio, A. Hanson, and E. Riseman. The schema system. *International Journal of Computer Vision*, 2:209–250, 1989.

[13] G. A. Fink. Developing HMM-based recognizers with ESMERALDA. In *Lecture Notes in Artificial Intelligence*, volume 1692, pages 229–234. Springer, 1999.

[14] N. Ghanem, D. DeMenthon, D. Doermann, and L. Davis. Representation and Recognition of Events in Surveillance Video Using Petri Nets. In *Second IEEE Workshop on Event Mining 2004, CVPR2004*, 2004.

[15] R. Gregor, M. Lützeler, M. Pellkofer, K.-H. Siedersberger, and E. Dickmanns. Ems-vision: A perceptual system for autonomous vehicles. *IEEE Trans. on Intelligent Transportation Systems*, 3(1):48–59, Mar. 2002.

[16] M. Hanheide, C. Bauckhage, and G. Sagerer. Combining environmental cues & head gestures to interact with wearable devices. In *Proceedings of the 7th International Conference on Multimodal Interfaces*, pages 25–31. ACM, Oct. 2005.

[17] N. Hofemann, J. Fritsch, and G. Sagerer. Recognition of deictic gestures with context. In *Proc. Pattern Recognition Symposium (DAGM)*, volume 3175 of *Lecture Notes in Computer Science*, pages 334–341, Heidelberg, Germany, 2004. Springer-Verlag.

[18] S. Li, M. Kleinehagenbrock, J. Fritsch, B. Wrede, and G. Sagerer. "BIRON, let me show you something": Evaluating the interaction with a robot companion. In *Proc. IEEE Int. Conf. on Systems, Man, and Cybernetics, Special Session on Human-Robot Interaction*, pages 2827–2834, The Hague, The Netherlands, October 2004. IEEE.

[19] D. Lowe. Distinctive image features from scale-invariant keypoints. *Int. Journal of Computer Vision*, 60(2):91–110, 2004.

[20] M. T. . A. Pentland. Eigenfaces for recognition. *Journal of Cognitive Neuroscience*, 3(1), 1991.

[21] J. L. Peterson. *Petri Net Theory and The Modeling of Systems*. Prentice Hall, Inc., Englewood Cliffs, Massachusetts,, 1981.

[22] W. Ponweiser, G. Umgeher, and M. Vincze. A reusable dynamic framework for cognitive vision systems. In *ICVS Workshop on Computer Vision System Control Architectures*, 2003.

[23] M. Ribo, M. Brandner, and A. Pinz. A flexible software architecture for hybrid tracking. *Journal of Robotics Systems*, 21(2):53–62, 2004.

[24] R. Rimey and C. Brown. Control of selective perception using bayes nets and decision theory. *Int. J. Comp. Vision*, 12(2/3):173–207, April 1994.

[25] H. Sidenbladh, M. J. Black, and D. J. Fleet. Stochastic tracking of 3d human figures using 2d image motion. In *6th European Conference on Computer Vision (ECCV 2000)*, pages 702–718, Dublin, Ireland, 2000. Springer Verlag.

[26] P. Viola and M. Jones. Robust real-time face detection. *Int. Journal of Computer Vision*, 57(2):137–154, 2004.

[27] W3C. XML-binary Optimized Packaging, W3C Recommendation 25 January 2005, 2005. `http://www.w3.org/TR/2005/REC-xop10-20050125/`.

[28] M. Weber and E. Kindler. The petri net markup language. In *Petri Net Technology for Communication Based Systems.*, LNCS 2472. Springer-Verlag, 2003.

[29] S. Wrede, J. Fritsch, C. Bauckhage, and G. Sagerer. An XML based framework for cognitive vision architectures. In *Proc. of ICPR*, number 1, pages 757–760, 2004.

[30] S. Wrede, M. Hanheide, C. Bauckhage, and G. Sagerer. An active memory as a model for information fusion. In *Int. Conf. on Information Fusion*, number 1, pages 198–205, 2004.

[31] S. Wrede, W. Ponweiser, C. Bauckhage, G. Sagerer, and M. Vincze. Integration frameworks for large scale cognitive vision systems - an evaluative study. In *Proc. of ICPR*, number 1, pages 761–764, 2004.