A Generic Context Information System for Intelligent Vision Applications

Luo Sun, Peng Dai, Linmi Tao, and Guangyou Xu

Tsinghua National Laboratory for Information Science and Technology
Tsinghua University, 100084 Beijing, China
{sunluo00,daip02}@mails.tsinghua.edu.cn,
{linmi,xgy-dcs}@tsinghua.edu.cn

Abstract. The future intelligent vision is expected to be highly context-aware such that it can perceive and be aware of user's situation and react accordingly. In this paper, we propose a context representation mechanism and build a high-performance, extensible, distributed context information system based on it, in order to facilitate context-awareness development and information sharing. It pays attention to representing and organizing contexual information in an effective way and does not force any certain type of context reasoning algorithm. It can provide information-related services for distributed intelligent vision applications, mainly including representation, storing and retrieval, forming a whole pipeline of real-time semantic metadata generating and management. Besides user context, which is used to support runtime context communication between application components, our system also contains contextual descriptions about running environment and system configuration, making applications based on it can move to another environment or configuration seamlessly. Moreover, context representation in our system has a well-designed plugin-based architecture, helping users add their own context types without any modification of the original system. We introduce a context-aware meeting application based on our system, which employs Dynamic Bayesian Network as context reasoning algorithm. Experiment results show our context information system has excellent configurability, extensibility and performance.

Keywords: Context Information System, Context Representation, Context Storing and Context Retrieval.

1 Introduction

The final goal of intelligent vision is to provide the right service to the right object in the right place on the right time, which is expected to be highly context-aware such that it can perceive and be aware of user's situation and react accordingly [1]. By context, we mean a dynamic structure of information that is used to characterize the situation, viewed over a period of time, episode of use, social interaction, internal goals and local influence [2, 3]. In the real world, understanding of each person's behavior is affected by his relations with others and the surrounding environment,

M. Bubak et al. (Eds.): ICCS 2008, Part I, LNCS 5101, pp. 759–769, 2008.

therefore contextual information has to be considered. Moreover, with the development of multimedia sensing technology, more and more data can be acquired by much less effort than even before. Facing large amount of available data, the use of context can help decrease the complexity significantly by processing only relevant scenarios and heuristics to choose only the objects that may involve in a particular activity.

The very first step of context-awareness is to represent context in an effective computer-applicable format [1, 2]. Besides representing contextual information, information representation and sharing is playing an important role in the domain of computer vision. Over last 20 years, we have seen a remarkable amount of progress in the abilities and usability of computer vision. Although there is lots of collaboration between groups working in the same field, the majority of researchers have their own ways of working and representation format. This causes the issue that we still only see very individual and proprietary use of this work, both in research and in industry [4]. In this regard, a generic context information system to facilitate context-awareness development and information sharing, which does not force any certain type of context reasoning algorithm, is the urgent need in the development of intelligent computer vision applications.

There have been several attempts for this purpose recently. Most of them represent information in XML because of its open, human-readable format and natural hierarchical structure. VEML (Video Event Markup Language) [5] has been implemented for representing and recognizing events in videos, in order to facilitate the development of applications such as video surveillance, video browsing and content-based video indexing. CVML (Computer Vision Markup Language) [4] summarized lots of common visual processing requirements and proposed a framework dedicated to describe computer vision information. MPEG-7, formally named "Multimedia Content Description Interface", is a wide-accepted standard for describing multimedia content data, which aims to resolve the problems of management and retrieval. It has rich functionality, which also makes itself too complex to employ. Moreover, MPEG-7 is not suitable for dynamic scenes. However VEML and CVML are not as wide-accepted as expected and these three information-description languages do not address the concept of context. Context Toolkit [6, 7] aims to be a reusable solution to handle context in a distributed infrastructure. However, it doesn't provide adequate support for organizing context in a formal structured format, therefore cannot represent the dynamic nature of context [3]. It also adopts hard coded context ontology and cannot be easily extended and interoperate with others.

Facing the issue that information systems don't consider context-awareness and context-aware systems don't take information sharing into account, we propose a context representation mechanism for intelligent computer vision and build a high-performance, extensible, distributed context information system based on it, in order to facilitate context-awareness and information sharing. It's used to support information-related services for distributed intelligent vision applications, including representation, storing and retrieval, forming a whole pipeline of real-time semantic metadata generating and management. We have already implemented a flexible multi-server platform for distributed visual information processing in our previous work [8]. Combining it with our context information system, a complete infrastructure of

intelligent computer vision has emerged. The rest of this paper is organized as follows: section 2 describes the system, including our basic idea, several designing aspects of the architecture and some key technologies. Section 3 shows a context-aware meeting application based on our information system, as well as some performance results. In the end, there will be some conclusions.

2 Context Information System

2.1 Distributed Architecture

As a complete information system, the following functions should be provided at least. They are minimal requirements to form a whole pipeline of metadata generating and management.

1. Information representation, the mechanism to formalize information into metadata, representing information in an effective computer-applicable way.
2. Information storing, the ability to store formalized information persistently, usually to hard disks. Information is usually indexed for faster retrieval.
3. Information retrieval, the ability to retrieve stored information later with specified restrictions.

Among them, information representation is usually considered as the most important one since it determines the representing scope and the ability of interoperation with others. However, as more and more sensors are employed into intelligent vision system, it's not possible to finish all procedures on just one computer. In this regard, a distributed architecture should be taken into account.

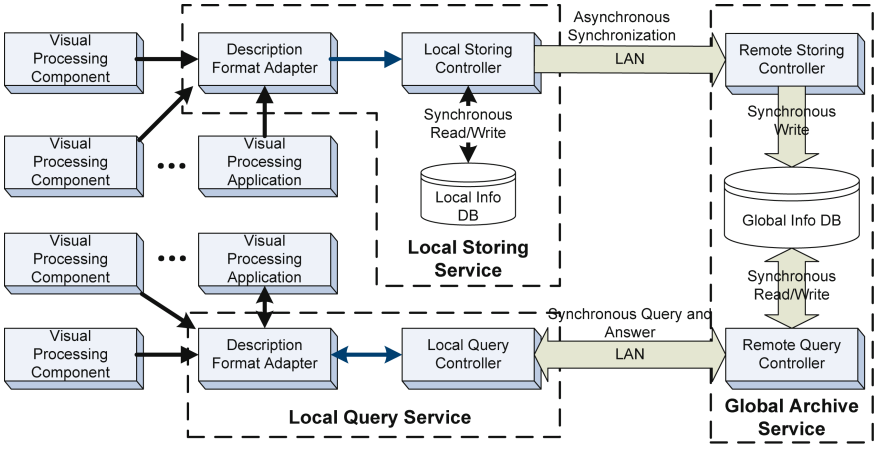

Fig. 1. System Architecture. The whole system is organized in a distributed manner, including three separate services, as shown in dash boxes.

In our previous work, a flexible platform for distributed visual processing has been finished, which acts as a container for information processing and analysis algorithms to plug in. The platform is composed by a set of servers which collaborate with each other to accomplish the tasks, such as video capture, transmission, buffering and synchronization. With the help of this convenient platform, we design a distributed architecture for our information system, as shown in Fig. 1. There are three kinds of services within the system, namely local storing service, local retrieval service and global archive service. Each service is a separate process, using local or remote socket to communicate with other services or visual processing components. Local storing service, local query service would run on every computer in the vision system, providing information-related services to all visual processing components working on the same machine. Global archive service is unique within the whole system, usually running on a dedicated server for information archiving.

We employ XML as basic storing format in our system. Therefore we can use Berkeley DB XML [9] to store metadata, which is a high performance, open source, embeddable XML database with XQuery support. It store XML in an optional indexed way, providing extremely fast access.

Our information system architecture features fast storing and retrieval access. This is important if want to deploy it to some existed intelligent vision applications, since they usually do not consider unified information format carefully at the very beginning and slow access may cause significant decreasing of their performances. Fast storing benefits from the local information database which is small and acts as local caching. New information will be stored into the local database directly by synchronized storing function calls, while there is a background thread dedicated to synchronize the local database with the global one. This design can effectively avoid the data loss in case of application crashes, while keeping storing procedure fast enough without annoying applications based on it too much. On the other hand, retrieval access cannot be handled in the same method. It has to be synchronized since applications need results immediately. We optimize it by the means of increasing the capacity of the internal cache of Berkeley DB XML and using UDP protocol with confirming replay, instead of TCP. The experiment later would show our context information system's performance.

2.2 Context Representation and Plugin-Based Implementation

The very first step of context-awareness is to represent it in a computer-applicable format. Traditional context models deal with individuals or their relationships with environmental objects, which can be represented as single level events such as changes of location, time, temperature etc. This method suffers especially under dynamic interaction scenes since it ignores the dynamic nature of context. Greenberg refined context as a dynamic construct, which is viewed over a period of time, episode of use, social interaction, internal goals, and local influence [3]. Following his definition, we propose a computer-applicable version of context representation in the domain of computer vision.

Context is defined as multi-level hierarchy, which represents situations at different abstract levels. Moreover, a complete context representation mechanism should not only include runtime dynamic information representing human physical and mental

states, but also need to contain static environment and configuration settings. There are three kinds of descriptions in our information system.

1. Environment configuration. The external description of a system, used to describe the environment where a system would run, including PC configuration, room configuration, physical objects and their properties, etc.
2. System configuration. The static internal description of a system, used to describe how a system would run, including the connection relationships between visual processing components, what kind of services the system should provide to users, etc.
3. Run-time information communication. The dynamic internal description of a system, used to regulate information format and share it through the whole system, helping all components understand and communicate with each other.

Respectively, context representation defined in our information system can be divided into three parts, environment context, system context and user context. User context is only valid during runtime and contains processing targets and their properties, from low-level features to high-level semantic descriptions. Environment context and system context are rather static, usually remaining unchanged during systems' running. User context highly depends on the interaction status, which changes frequently. It's organized in a hierarchical structure at different time scales and group sizes. Fig. 2 shows a user context example under meeting scenario.

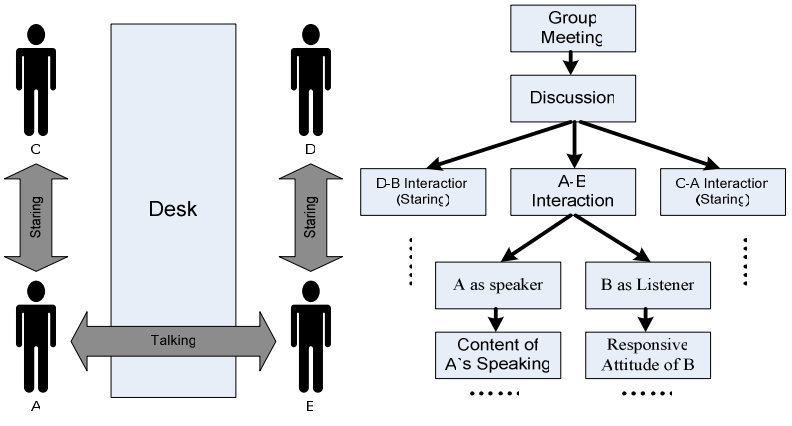

Fig. 2. A User Context Example under Meeting Scenario. There are four people within a meeting. At one moment, A and B is talking to each other, while C is staring at A and D is staring at B. Some key nodes of the current user context are showed as a tree on the right side. At the longest time scale, it's a discussion during a group meeting. There are three short-term interactions currently, namely A talking to B, D staring at B and C staring at A.

Context representation in our system is implemented in a plugin architecture. Plugin-based systems are widely used recently, e.g. Firefox and Eclipse. It has lots of advantages, such as rich extensibility and easy maintenance. It also helps reduce

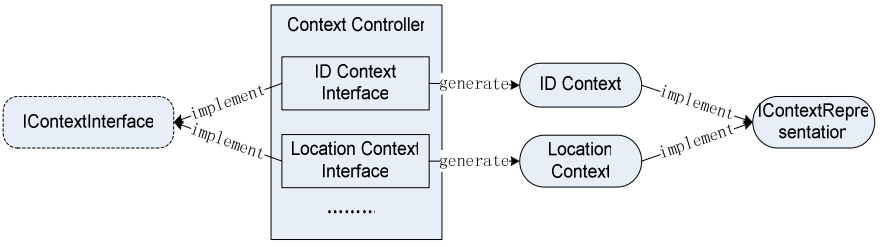

Fig. 3. Modules in Our Plugin System. ID context plugin, which has implemented interface *IContextInterface*, is a DLL(Dynamic Link Library) file under Windows or a SO(Shared Object) file under Linux and can produce ID context representation, which has implemented interface *IContextRepresentation*. Location Context component is the same. All plugins are controlled by a module named *Context Controller*, which provides several useful functions for plugin management, such as loading, refreshing and searching for a specified plugin.

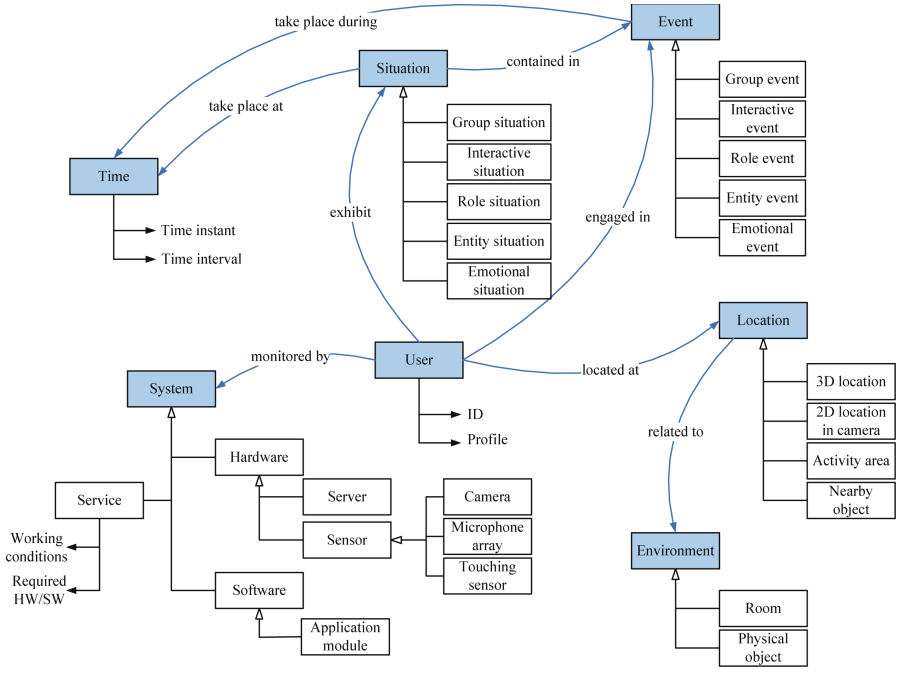

Fig. 4. Some of Implemented Context Representation in our Information System. Blue rectangles are categories and blue arrows between categories represent their relationships and how they work together. Each category contains several context types, which can be used to describe objects or theirs properties in the real world. Expect category “System” and “Environment”, all categories belongs to user context.

future development labor to some extent. In our design, each context type belongs to a

certain category, which is used to organize context types based on what kind of information they designate. Context category is actually a directory on the disk with a configuration file to describe its relationship with other categories. Codes of every context type are stored as a Dynamic Link Library file (DLL) under Windows, and a Shard Object file (SO) under Linux, in the corresponding category directory. When the information system starts, it will scan all possible directories and load context representations dynamically. Therefore, users can add their own context representations, just following some predefined interfaces, without compiling the whole information system.

The implementation of our plugin system is based on Qt's plugin system [10] and the relationships among modules are show in Fig. 30. Every context plugin needs to implement a common interface *IContextInterface* and every context representation needs to implement a common interface *IContextRepresentation*.

Some of implemented context representations in our system are shown in Fig. 4. It contains many useful context types for intelligent vision. Although they may not cover every requirement, developer can add their own context types very easily with the help of our plugin architecture.

3 Experiment Systems and Results

In this section, we would like to show a context-aware meeting application supported by our context information system, as well as some results of performance tests.

The major task of this indoor meeting analysis and archiving system is to generate and archive a hierarchical context representation of multimodal meeting data, which is can be further employed for retrieval and more sophisticated analysis [11]. Our

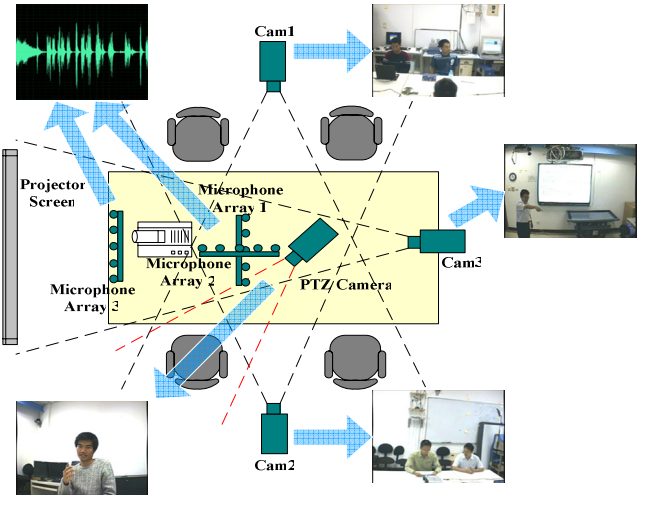

Fig. 5. Sensor Settings in Meeting Room. Three fixed cameras are set to monitor the meeting room from three distinct perspectives. A PTZ camera is placed on the table to focus on any

specific target. Three linear microphone arrays are assembled on the table to get audio information from various participants.

information system plays an important role in environment configuration, system configuration and user context representation. A flexible multi-server platform has been developed to support distributed multimedia data capturing and transferring. Multiple audio and visual sensors are installed in the meeting room so as to extract multimodal information in real-time, as illustrated in Fig. 5.

Since context cannot be determined by simple collection of multimodal sensor data, this system employs Dynamic Bayesian Network for context reasoning, which can generate analysis results at run-time. The structure of user context is shown in Fig. 6, where different layers have very strong semantic meanings. Fig. 7 shows a description example. At one moment, person A is staring at B (not shown in the picture). Some body parts of A, including head position and orientation, hands positions and so on, are tracked as low-level features for later analysis.

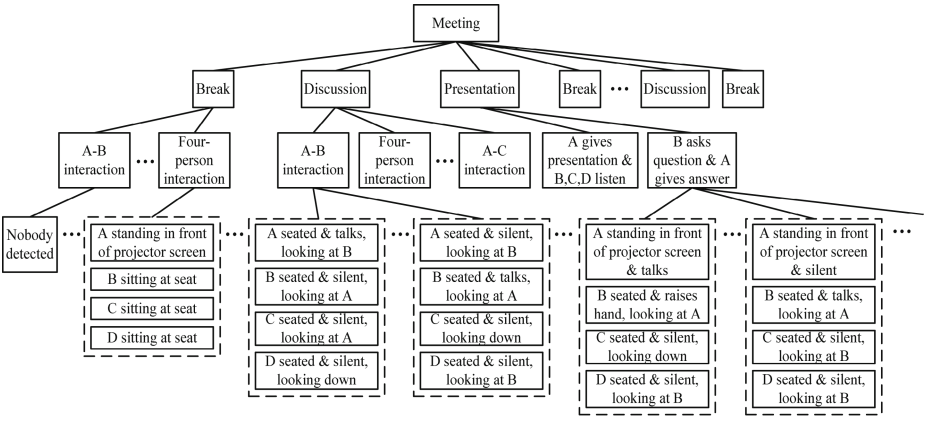

Fig. 6. Hierarchical User Context Structure under Meeting Scene. The top layer is the meeting scenario itself. The second layer is used to describe what current situation is from a high semantic perspective. The third layer represents how people interact with each other individually.

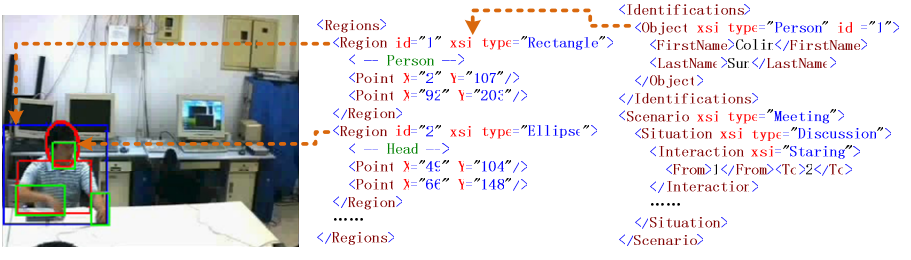

Fig. 7. A Description Example under Meeting Scene. The picture on the left side is from the original video overlapped with some color boxes designating different parts of a person. The

XML segments in the center and on the right are the corresponding low-level and high-level semantic descriptions, respectively. Orange arrows help show the relationships among them.

Storing and retrieval performance have been tested on data extracted from several meetings and another context-aware outdoor surveillance application. Fig. 8 shows some results about storing performance. From the figure, we can find that local storing speed is fairly fast, less than 2ms even when the local database is 20MB. However the storing speed is affected by the size of local database and the concurrency. Synchronization time increases with the size of global database and local database, nearly linearly.

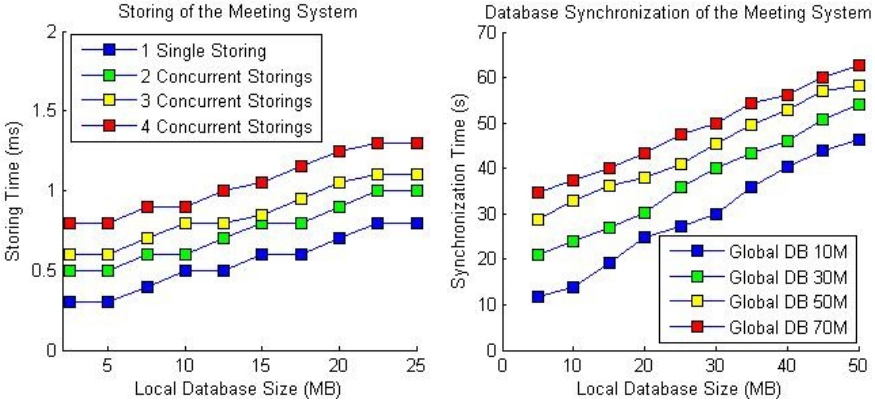

Fig. 8. Results of Storing Performance Experiment. The left plot shows some results of storing a specified event into the local database and different square colors represent different numbers of concurrent storing. The right one shows the background synchronization time and different square colors represent different sizes of global database.

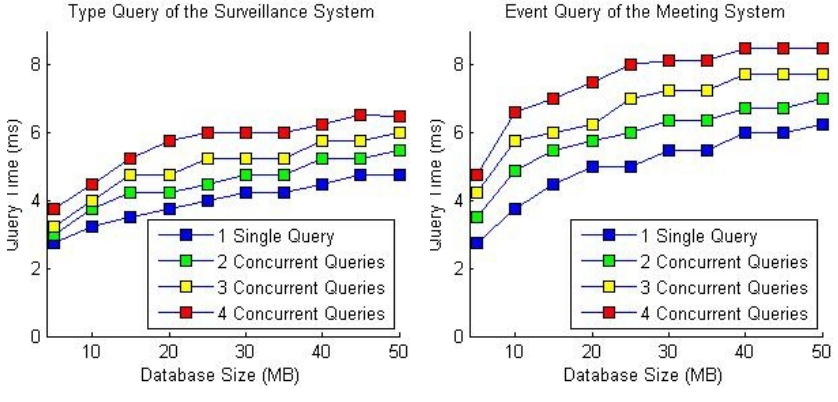

Fig. 9. Results of Retrieval Performance Experiment. These figures show how query time changes with respect to database size and concurrency. The left plot is searching for a specified type in the surveillance application, while the right one for a specified event in the meeting application. Different square colors represent different numbers of concurrent queries.

Fig. 9 shows some experiment results about retrieval performance. Event query is a little bit more complex than type query; therefore it costs a bit more time. Query time increases with the size of database and the number of concurrent query processes sublinearly. When database is small, query is fast since cache could be very hot, playing an important role in retrieval. With the increasing of database size, disk I/O costs more and more time.

4 Conclusion

Our work in this paper focused on the reusability of context by the means of splitting it into representation and reasoning. Reasoning differs significantly through different context-aware systems while representation remains similar. As a proof of concept, we presented a context representation mechanism and build a high-performance, extensible, distributed context information system based on it to facilitate context-awareness and information sharing. It can support information-related services for distributed intelligent vision applications, mainly including representation, storing and retrieval, forming a whole pipeline of real-time semantic metadata generating and management. Our proposed contextual information includes not only static environment and system settings but also dynamic information representing human physical and mental states. The former is used to describe where and how the application would run, while the latter is used for runtime intercommunication between application components. Contextual information has been implemented in a plugin manner, which means developers can add their own context types without any modification of the original system, just following our interface definition. As a result of our basic idea, this context information system pays attention to representing and organizing contexual information in an effective way and does not force any certain type of context reasoning algorithm; therefore developers can employ any algorithm they prefer, rule-based or probability-based. We also introduce a context-aware meeting analysis and archiving application based on our system, which employs Dynamic Bayesian Network as context reasoning algorithm. Our information system plays an important role in it, showing its configurability and extensibility. Performance of storing and retrieval has been tested on lots of real data and results show our system has excellent performance. In our previous work, we have already finished a flexible platform for distributed visual processing. Our information system actually can be considered as an important complement to it. Combining them together, a complete infrastructure of intelligent vision has emerged.

Acknowledgments. This research is supported by NSFC project 60433030 and 60673189 of China. We are about to be thankful for the thoughtful comments and suggestions of our reviewers.

References

1. Goh, E., Chieng, D., Mustapha, A., Ngeow, Y.-C., Low, H.-K.: A Context-Aware Architecture for Smart Space Environment. In: Proceedings of International Conference on Multimedia and Ubiquitous Engineering, Korea, pp. 908–913 (2007)

2. Dey, A.K., Abowd, G.D., Salber, D.: A Conceptual Framework and a Toolkit for Supporting the Rapid Prototyping of Context-Aware Applications. Human-Computer Interaction 16, 97–166 (2001)
3. Greenberg, S.: Context as a Dynamic Construct. Human-Computer Interaction 16, 257–268 (2001)
4. List, T., Fisher, R.B.: CVML – An XML-based Computer Vision Markup Language. In: Proceedings of International Conference on Pattern Recognition, Cambridge, vol. 1, pp. 789–792 (2004)
5. Nevatia, R., Hobbs, J., Bolles, B.: An Ontology for Video Event Representation. In: Proceedings of International Conference on Computer Vision and Pattern Recognition Workshop (2004)
6. Context Toolkit, http://www.cs.cmu.edu/~anind/context.html
7. Edwards, K., Bellotti, V., Dey, A.K., Newman, M.: Stuck in the Middle: The Challenges of User-Centered Design and Evaluation for Middleware. In: Proceedings of The International Conference on Human Factors in Computing Systems (2003)
8. Wang, Y., Tao, L., Liu, Q., Zhao, Y., Xu, G.: A Flexible Multi-server Platform for Distributed Video Information Processing. In: Proceedings of 5th International Conference on Computer Vision Systems (2007)
9. Oracle Berkeley DB XML, http://www.oracle.com/database/berkeley-db/xml/index.html
10. Qt product page, http://www.trolltech.com/products/qt
11. Dai, P., Di, H., Dong, L., Tao, L., Xu, G.: Group Interaction Analysis in Dynamic Context. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics (to appear, 2007)