DiVA: a Distributed Video Analysis framework applied to video-surveillance systems

Juan Carlos San Miguel, Jesús Bescós, José M. Martínez and Álvaro García

Grupo de Tratamiento de Imágenes
Escuela Politécnica Superior, Universidad Autónoma de Madrid, SPAIN
E-mail: {Juancarlos.Sanmiguel, J.Bescos, JoseM.Martinez, Alvaro.Garciamartin}@uam.es

Abstract

This paper describes a generic, scalable, and distributed framework for real-time video-analysis intended for research, prototyping and services deployment purposes. The architecture considers multiple cameras and is based on a server/client model. The information generated by each analysis module and the context information are made accessible to the whole system by using a database system. System modules can be interconnected in several ways, thus achieving flexibility. Two main design criteria have been low computational cost and easy component integration. The experimental results show the potential use of this system.

1. Introduction

The growing demand for surveillance, especially for outdoor/indoor security in buildings, is increasing the need to create and develop intelligent systems. Most surveillance systems suffer from non-scalability or low frame rates due to computationally expensive algorithms. Current research in this field is focused towards the design of wide-area automatic and intelligent surveillance systems (third generation systems)[1]. These systems can be categorized as concurrent, distributed, embedded and real time. They are usually designed in a synchronous way using an object-oriented approach (e.g. CORBA), what can produce bottlenecks or exhaustion of resources.

Usually, the systems deployed are designed for a specific context and they can't be easily adapted to other contexts. It is generally accepted that the use of context information allows to improve the quality results of the analysis processes[2]. Traditionally, this information has been incorporated into these processes through manual parameters adjustment or implicit inclusion in the algorithm code.

The present work proposes an approach to the design of intelligent distributed surveillance systems. This approach is based on a distributed and flexible configuration design adding low computational cost to the algorithms running on this system. Due to easy component integration, the system provides a good environment for researchers to develop new video analysis algorithms.

The paper is organised as follows: section 2 describes the state-of-the-art in video-surveillance system design and architectures, section 3 and 4 describe, respectively, the system and its components, section 5 shows the performance obtained with this flexible and scalable approach and finally section 6 concludes and indicates future work.

2. State-of-the-art

The work presented in this paper covers mainly the design and implementation of video-analysis systems where multiple cameras are in play; video-surveillance systems are currently one of the most important ones demanding these features. The requirements for designing this type of systems are the object of very active research[1][3][4]. In general, they can be described by the following desirable functionalities: 1) scalable systems with computational charge distribution, 2) real-time operation, 3) low resource consumption, 4) communication control, 5) communication over standard networks and 6) runtime re-configuration.

Several surveillance systems have been designed and developed by industry and academia. In the literature, these systems are described on a very high level not allowing the distinction among the functionalities of the different approaches. One of the classical distinctions between these systems is their purpose: they can be divided into specialized and general-purpose systems. For instance, [5] focuses on metro stations and [6] focuses on traffic surveillance.

Another classification can be distributed and non-distributed systems. The need of designing distributed systems is briefly discussed in [4]. The distribution process can be further categorized depending on which technology is used. Thus, there are systems like [7][8] that use their own communication protocol to distribute work into the network; others use RSTP[9], SOAP[4] or CORBA[5] to manage their communications. Systems can be also classified based on the existence or not of a centralised server to control all the system components. In [3] a system without server is presented, that avoids the centralisation, making all the independent subsystems completely self-contained. On the other hand, systems like [10] use a central server that restricts the system scalability, allowing a better system management.

Usually the design of these types [5][6] of systems is based on an object-oriented and synchronous approach (e.g. CORBA). This approach can produce overhead at run-time and may cause communication bottlenecks. To avoid this constraint, the MASCOT[11] design method was proposed for improving the communication protocol making it simple and allowing asynchronous communications.

3. Framework overview

The proposed framework is divided in two levels of abstraction: physical part and logical part.

The physical part (see Fig. 1) is composed of the required hardware: the cameras and a cluster of standard computers (PCs) connected together through a fast Ethernet network.

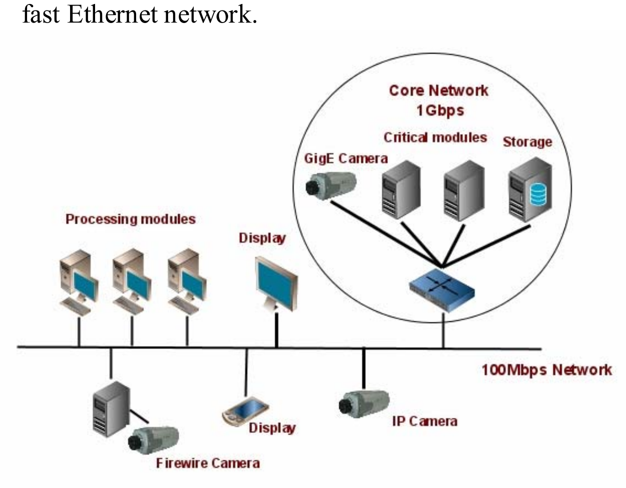

Figure 1. Physical system description

In order to cope with bandwidth restrictions and to improve the system performance, the network architecture is composed of two networks. The main processing units of the system are a set of rack-mounted standard PCs interconnected by a dedicated Gigabit Ethernet (core network). The other system units (mainly processing modules) are distributed in a 100BaseT Ethernet network around the core network. Different types of cameras are plugged either to an acquisition card on a PC or directly to the Ethernet network for IP cameras. The computers are used to acquire the video, run algorithms and store the data. The main advantage of this architecture is the flexibility. Future needs in computing power will be simply addressed by adding PCs (or replacing existing ones with more powerful ones) in the cluster.

The logical part is composed by four independent layers (see Fig. 2). Each layer is designed in a modular way and has an specific role. The next section describes them with more detail. The different modules can be distributed in several ways allowing flexible configuration. Also the system supports the addition of modules at operation time. Depending on application requirements, layers can be combined into one single component with the required functionality (e.g. fast data acquisition, fast module intercommunication).

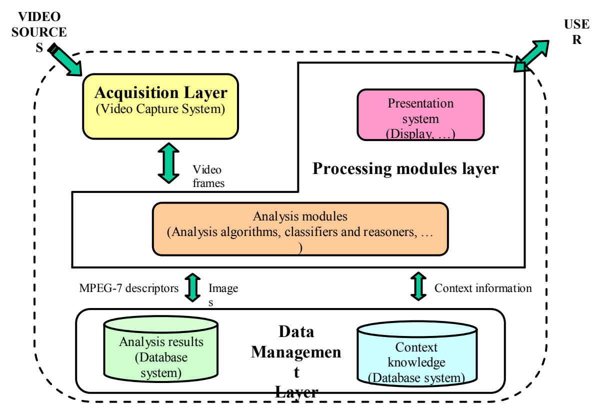

Figure 2. Logical system layers

The main features of the proposed system are:

- Distributed environment for research, prototyping and deployment of visual analysis systems with multicamera and contextual information support.
- Modular and multithreaded design.
- Based on processing at frame level.
- Based on a client/server model.
- Flexible configuration (cascading or parallel interconnection of processing algorithms).
- Monitoring and reuse of analysis results.
- Plug and play support .
- Asynchronous operation mode.

4. Logical System layer description

This section describes the different layers of the logical system of DiVA.

4.1 Acquisition layer

This layer acquires the video from multiple video feeds and distributes video frame-by-frame to the whole system. The distribution is based on a server/client model. Video frames are currently exchanged using baseline JPEG (ISO/IEC 10918-1) or uncompressed format. A time stamp is attached to each frame at grabbing time, and used in the processing stage (e.g., tracking algorithms).

Due to its modular design, the system can easily support new camera connectivity protocols with very few effort. Currently, our system handles IP, IEEE1394, GigE and USB protocols as well as input via video files.

4.2 Communications layer

The communication between system components is based on a server/client model using point-to-point; the flow control is realized through a TCP-based network. To initiate a communication, the client logs on the server and then it starts the data transfer. The communications protocol is based on transmitting only useful information, allowing fast communication between modules. To avoid network problems, data buffering between modules is supported at both sides.

4.3 Processing modules layer

In our approach, a processing module is a component of the system responsible for some particular task not related to the other layers (e.g. video content analysis module, player module).

The modules run concurrently and asynchronously: typically each module will run on its own processor, but this is not mandatory. This is the main mechanism of parallel and distributed processing in our approach. The modules are arranged in a fixed network topology as determined by the overall system structure. They communicate only with Acquisition and Data Management layers to request and store data. In these two layers, there are servers that are waiting to deliver the content stored. The processing modules act as clients requesting the video data and analysis results necessary for completing their task. Each module has a specific task, and the modules cooperate to achieve the system goals. The communication between modules is mapped through a database system. Modules read/write their results in the database system (acting as a content server).

To allow easy integration in the overall system, some module templates have been created to ensure fast development of new algorithms within the framework.

4.4 Data Management layer

This layer is in charge of storing and distributing information required for analysis purposes. Due to the existence of different types of information in the system, this layer is divided into two modules: one for analysis results and other for contextual information.

The first module is composed of a database that manages the availability and intercommunication of analysis results between system modules. This module has two different levels of storage: one for system configuration and the other for analysis results. Metadata associated to analysis results are generated in MPEG-7 format. To exploit the information described, the metadata are extracted from the MPEG-7 structures and stored in the database. A relational database is used with a straightforward mapping of the data onto database tables. Additionally some of the MPEG-7 structures are compressed to reduce their size providing a bandwidth reduction in the communication processes.

The second module enables the use of domain and application context knowledge in the whole system. It allows the use of the system (designed as a ‘generic’ system) in different specific scenarios. This kind of information is coded using ontologies for knowledge representation and stored in this module. Then, system modules can request this information and use it in their tasks.

5. Results and performance

The proposed framework has been validated on real use-cases. One of them is detection of abandoned or removed objects in three different scenarios (each one with one camera input; IEE1394, IP and GigE) at the same time, alerting when an event occurs.

A logical description of the system modules can be seen in Fig. 3. The implementation runs over a Pentium 4 at 2.8GHz under Windows OS. For each scenario, the system extracts the foreground objects and classifies the static foreground regions as abandoned or removed objects. The candidate foreground object extraction task involves two Processing Modules (PM): one performs foreground segmentation (PM1) and the other candidate object extraction (PM2). The former generates a binary mask that is stored in the database system. The latter requests video frames and previous segmentation masks, and generates an MPEG-7 description of the candidate objects that is stored in the database system. Finally, module PM3 performs event detection using video data and the MPEG-7 description previously generated: candidate objects are classified as abandoned or removed by matching the boundaries of static foreground regions. These three modules operate in the same PC.

Module PM4 uses the final MPEG-7 descriptions generated for each scenario, showing the alarms detected over the corresponding scenario frames.

Measured metadata flow is less than 15KB/s per communication. It shows that the overhead introduced by multiple transmissions is lower than a video frame transmission.

Overall results show that the system can process data from multiple cameras at real-time at CIF resolution, confirming that the computational cost introduced by the system does not reduce its performance. This configuration demonstrates that DiVA allows flexible configuration.

6. Conclusions and future work

This paper presents a distributed surveillance system/framework that allows flexible configuration and dynamic reconfiguration at runtime. Due to low computation management cost, it operates at real-time over standard computers. Additionally, it provides a flexible environment to develop computer vision algorithms via easy component integration.

For the future, other extensions and improvements will be made on the global system, like integration of a compressed video analysis path or adaptation of the system to work under Linux (using POSIX threads for multitasking scheduling).

7. Acknowledgements

This work is supported by Cátedra Infoglobal-UAM for “Nuevas Tecnologías de video aplicadas a la seguridad”, by the Spanish Government (TEC2007-65400 SemanticVideo), by the Comunidad de Madrid (S-050/TIC-0223 - ProMultiDis-CM), by the Consejería de Educación of the Comunidad de Madrid and by The European Social Fund. The authors would like to thank J. Molina, H. Cabrejas and V. Fernández-Carbajales for their valuable contributions.

References

[1] M. Valera and S.A. Velastin, “Intelligent distributed surveillance systems: A Review”, IEEE Procs. of VISP, 152(2):192-204, 2005

[2] N. Mallot, M. Thonnat, A. Boucher, “Towards Ontology Based Cognitive Vision”, Procs. of MVA, 16(1):33-40, 2004.

[3] M. Christensen and R. Alblas, "V2-design issues in distributed video surveillance systems," Tech. Rep., Department of Computer Science, Aalborg University, 2000.

[4] R.G.J Wijnhoven et al, “Flexible Surveillance System Architecture for prototyping Video Content Analysis Algorithms”, Procs. of SPIE 6073, 2006.

[5] N. Siebel and S. Maybank, "The advisor visual surveillance system", Procs. of ECCV, pp.103-111, 2004.

[6] S. Kamijo, et al.“Development and evaluation of real-time video surveillance system on highway based on semantic hierarchy and decision surface", IEEE Procs. of SMC, pp. 840-846, 2005.

[7] X. Yuan et al, “A distributed visual surveillance system”, Procs. of AVSS, pp. 199- 204, 2003.

[8] H. Dias et al, “Distributed Surveillance System”, Procs. of EPIA, pp. 257-261, 2005

[9] D. Ostheimer, “A modular distributed Video Surveillance System Over IP”, Procs. of CCECE, pp 518-521, 2006

[10] B. Lo et al, "An Intelligent Distributed Surveillance System For Public Transport", IEEE Procs. of IDSS, pp. 10/1-10/5 , 2003

[11] M. Valera et al, “Real-Time Architecture for a large distributed Surveillance System”, IEEE Procs. of IDSS, pp 41-45, 2004

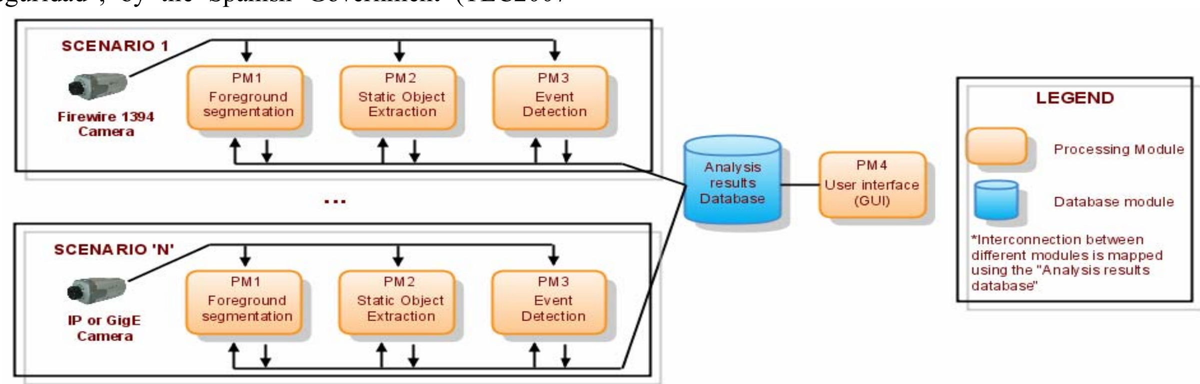

Figure 3. Logical system description for sample application