BUILDING MODULAR SURVEILLANCE SYSTEMS BASED ON MULTIPLE SOURCES OF INFORMATION

Architecture and Requirements

Daniel Durães, Luís F. Teixeira and Luís Corte-Real
INESC Porto, Faculdade de Engenharia, Universidade do Porto
Campus da FEUP, Rua Dr. Roberto Frias, no. 378 4200-465 Porto, Portugal

Keywords: Intelligent Surveillance Systems, Modular Architecture, Multi-sensor surveillance, Data Analysis.

Abstract: Intelligent surveillance is becoming increasingly important for the enhanced protection of facilities such as airports and power stations from various types of threats. We propose a surveillance system architecture based on multiple sources of information to apply on large scale surveillance networks. The main contribution of this paper is the definition of the requirements for a flexible and scalable architecture that supports intelligent surveillance using, alongside video, different sources of information, such as audio or other sensors.

1 INTRODUCTION

Visual surveillance systems are widely applied in transport scenarios, such as airports, railways, underground, and motorways as well as other public spaces, such as banks and shopping malls. Typically these systems are manually handled by a human operator and do not rely on content-based automation processes. The captured content is stored for a limited period of time, due to storage limitations. Also, searching for a specific event can be a very time consuming task. Recent advances in this area are opening new possibilities for the next generation surveillance systems (Valera and Velastin, 2005). The main focus of research is on improving image processing tasks by generating more accurate and robust algorithms for object detection and recognition, tracking and human activity recognition.

A new generation of systems for surveillance are also starting to be commercialised. The common processing tasks that commercial systems perform are intrusion, motion detection and packages detection. Typical examples of commercial surveillance systems are DETEC [1] and Gotcha [2]. They are usually based on what is commonly called motion detectors, with the option of digital storage of the detected events. These events are usually triggered by objects appearing in the scene. Another example of a commercial system intended for outdoor applications is DETER (Pavlidis et al., 2001), which reports unusual movement patterns of pedestrians and vehicles in outdoor environments such as car parks. A broader goal is defined by the PRISMATICA project (Lo et al., 2003). The developed system is a wide-area multisensor distributed system, receiving inputs from CCTV, local wireless camera networks, smart cards, and audio sensors. Also, the project ADVISOR (Siebel and Maybank, 2004) aims to assist human operators by automatically selecting, recording, and annotating images containing events of interest. ADVISOR interprets shapes and movements in scenes being viewed by the CCTV to build up a picture of the behavior of people in the scene.

In summary, intelligent surveillance systems are not restricted to cameras but can also contain different types of information sources, to better interpret the danger. Monitoring surveillance networks by human inspection is expensive and ineffective. Consequently, surveillance users are choosing software for automated video surveillance. The architecture presented, proposes a solution to integrate specific surveillance algorithms, acquire information from various sources and interact with external systems. The solution presented, specifies architecture organized in a hierarchical structure and divided into different modules, including both support for communication and for computation.

[1] http://www.detec.no

[2] http://www.gotchanow.com

Durães D., F. Teixeira L. and Corte-Real L. (2008).
BUILDING MODULAR SURVEILLANCE SYSTEMS BASED ON MULTIPLE SOURCES OF INFORMATION - Architecture and Requirements.
In *Proceedings of the International Conference on Signal Processing and Multimedia Applications*, pages 314-319
DOI: 10.5220/0001936803140319

2 SYSTEM REQUIREMENTS

Our objective is to devise a software architecture for surveillance systems composed of a sensor network of different sources. The architecture should allow adding new and improved surveillance techniques while the network continues operating. The requirements are divided in non-functional, functional and hardware categories. The first two categories of requirements are based on (Detmold et al., 2006) and (Valera and Velastin, 2008).

2.1 Non-Functional Requirements

The non-functional requirements for video surveillance networks include scalability, availability, evolvability, integration, security and manageability.

- *Scalability*: We should consider different scopes for scalability. Processing the data generated in a large-scale surveillance system in a centralized architecture is unfeasible and a scalable distributed processing is required. Also, storing the relevant data, even if a small fraction of all captured data, requires a scalable storage distributed system. Finally, in some cases a scalable network needs to be considered, mainly if the number of sources of information are expected to increase.
- *Availability*: A larger number of components increases the probability that some component will eventually fail. However, the architecture must support acceptable levels of operation. A configuration of redundant systems should be considered if high level of availability is required.
- *Evolvability*: Within some limits, the surveillance network should be able to accommodate changes, including alterations to the hardware and changes to the software.
- *Integration*: Nowadays surveillance systems usually operate in independently of other systems. However, there is a growing need to integrate different systems, especially for intelligent management of buildings. It is therefore required a perspective of integration with external systems.
- *Security*: The integration with external systems however intensifies the need for security. A critical requirement is that the system should be robust to attacks with malicious intents. The consequences of such attacks include compromising confidential information or shutting down the system through denial of service attacks.

2.2 Functional Requirements

The functional requirements of intelligent surveillance systems include modules that perform: (1) signal processing, including audio and visual processing, (2) data aggregation and higher-level semantic analysis, (3) command, control and inspection of all network elements, and (4) storage with browsing and forensic analysis capabilities.

We first consider a low-level *signal processing* module that receives data from sensors and generates a sparser representation of data when compared to the raw data. For example, for the visual sensors, i.e. cameras, the signal processing module includes: *object detection*, *object classification* according to shape, color, and other properties, and *object tracking* along time within individual cameras views. In this case, the signal processing module accepts images from individual cameras as input, and produces as output a compact representation of the content – for example, the object trajectories. These modules are closely related to each sensor and can be seen as a layer just above the sensors. Depending on the hardware capabilities of the sensors, the modules' implementation could be embedded in the sensors.

The data processed for individual sensors is collected by a *data aggregation* module that relies on the multiple sources of data in the network to get a more complete representation. Using the same example, for visual sensors the data aggregation module includes: *multicamera tracking of objects* across time and multiple cameras views. Also, a *higher-level analysis* is performed to extract semantic knowledge, which usually requires a priori information and training. One objective could be *behavioral analysis*, or in other words, to recognize and understand the activities of the tracked objects. The output of these modules is typically auxiliary data (or metadata) that is stored alongside the raw data or events to alert the human operator.

The events triggered by the system warn about situations that eventually require close attention by the operators. When needed, the operators use *command control, and inspection* functionalities to interact with system – e.g. examine the status of an element (e.g. a sensor) in the network and take corrective actions. Other examples of interactions include remote *sensor control* of individual sensors in real-time, *target following* by selecting an object to follow and making a report, and *external system control* namely, controlling the elevators system or locking the doors.

Finally, *storage with browsing and forensic analysis* capabilities allow an operator to efficiently find a given event or object. Browsing or querying the

historical archive, the operator can look for detected events and know the details of possible threats. The auxiliary data mentioned previously is used to aid browsing and provide forensic analysis. Exploring large amounts of data without such capabilities can be extremely time consuming and inefficient.

2.3 Hardware Requirements

This architecture proposes the sharing of data though different interfaces. The interfaces depend on the needs of the system but typically the whole hardware is connected by a local area network (LAN).

The system is composed of computers or smart sensors (mostly cameras) connected by the LAN. A human-computer interface and a storage farm are also plugged in this system. The type of sensors that are possible connect besides cameras and microphones are smoke sensors, biometric sensors among others.

The storage module needs to allow 24 hours/day of data. Since only relevant data is stored, the amount of data produced in a day may vary. If we use MPEG-4 technology at 25 fps and 160 Kbit/s, in 3 days the total amount of raw data for a camera is 40 GB. However only a much smaller fraction of this needs to be stored ($\sim 5\%$).

A distributed system implies an efficient use of the available bandwidth that satisfies quality of service (QoS).

3 ARCHITECTURE

The concept of the architecture is based on a hierarchical relation between elements, respecting functional and hardware requirements and allowing interaction with external systems. Figure 1 shows a physical view of the surveillance system architecture using the standard UML model representation.

The main modules in the architecture are *control and command*, *database centre* and *monitoring*. All these modules contain a component called *communications manager*. Its goal is to manage the communications between different modules.

3.1 Control and Command

With the control and command module, human operators can visualize the outputs of the system and eventually take an action. The interaction between operators and the system is done through a graphical user interface (GUI) is possible interact with system.

A possible implementation solution is to use a web-based GUI, implemented in PHP or Java (server-side) and HTML with Javascript (client-side).

3.2 Database Centre

The database centre module is a physically independent module, where storage all data produced by system. This architecture is both flexible, because the database centre module is independent of other modules, and scalable, because many database centres may operate together or separately. The redundancy provided by multiple database centres is also essential to maintain high availability of operation.

In (Black et al., 2004), the authors present a solution that propose a creation of a database with four layers, ordered hierarchically, supporting high and low level queries. The data is stored selectively by the different layers. The system provides a mechanism to generate video content summaries of objects detected by the system across the entire surveillance region in terms of the semantic scene model. The layers of abstraction are: (1) raw data layer, (2) object motion layer, (3) semantic description layer, and (4) metadata layer. At the lowest level the system user can manually browse a stored video to observe some object activity recorded by the system. At the highest level, queries could be executed to automatically find that same object activity.

3.3 Monitoring

This is the system module closer to the hardware. Monitoring is performed independently for each surveillance sensor and afterwards aggregated by a multimodal data analysis system. The monitoring module contains the following components:

- *Signal processing, including visual and audio analysis*: Processes the data captured by the distributed network of sensors. For the visual sensors it typically includes object detection, classification and tracking. A similar flow is done for audio processing, including audio source separation and classification to detects dangerous situations, such as screams or explosions. If an array of microphones is available, localization and tracking of audio sources is also possible. Each of the components (visual analysis and audio analysis) is associated to a single sensor.
- *Analysis of data from other sources*: Analyses the information obtained by other sources, such as emergency door sensors, fire detection alarms, etc.

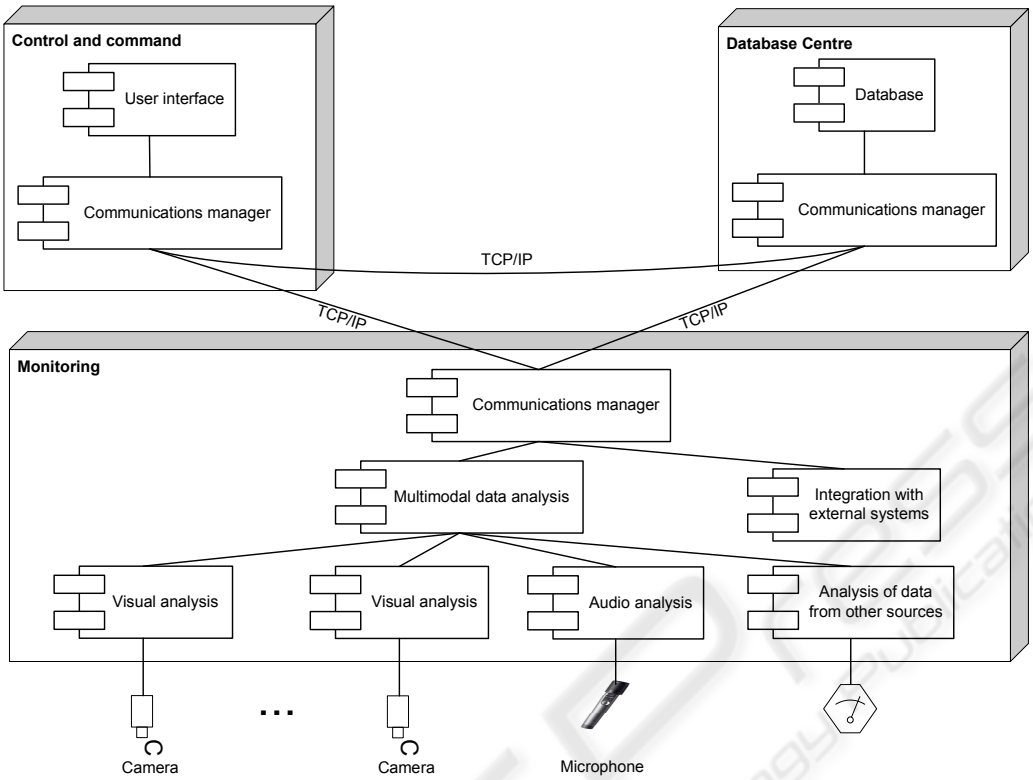

Figure 1: Component diagram of the system architecture.

- *Multimodal data analysis*: Receives data from the signal processing and the analysis of data from other sources components. Performs data aggregation and fusion, such as multicamera tracking and behavioral analysis. The processed data and generated events are sent to the storage module. If a give event is configured to generate an alarm, the human operator is notified by the control and command module.
- *Integration with external systems*: Interfaces with external systems and executes the orders given by human operator, for example building access.

With this architecture based in modules and components it is possible to capture different types of information and analyse them to produce better results when compared to the results obtained by a separate analysis of sensor information. The ultimate goal is to minimize the errors generated by the system.

The components performing data analysis (including visual, audio, other sources and multimodal analysis) are typically computationally intensive and require a low-level implementation in C or C++. The integration of individual components can be implemented using higher-level programming languages, such as Perl and Python.

4 IMPLEMENTATION

The system is currently being developed and not all specified modules are fully functioning. In this section we give an overview of the current status of implementation.

4.1 Algorithms

Object detection is done using cascade algorithm (Teixeira et al., 2007) essentially based on mixture of Gaussians for background modelling (Lee, 2005). Taking in consideration that background changes are caused by phenomena of different nature, a cascaded evaluation of typical dynamic elements is performed. These elements include acquisition noise, illumination variation and slow or repetitive structural changes. The latter type of changes is classified using the methods that estimate a p.d.f. to model the background. Also, a statistical test and a simple collinearity test are used to classify changes originated by noise and illumination changes, respectively. Some results are shown in Figure 2.

Tracking of objects is done using an implementation based on (Zhao and Nevatia, 2004). This track-

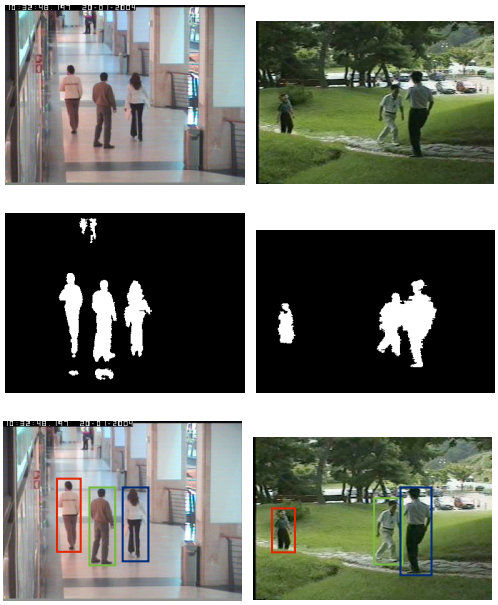

Figure 2: Segmentation results (second row) using the cascaded change detection algorithm and tracking results (third row) using the Kalman filtering with appearance constraints.

ing method relies on Kalman filtering for hypothesis tracking, and on the objects appearance constrained by its shape.

Tracks from different objects are matched to find multiple appearances of an object as in Figure 3. This is accomplished with a combined representation scheme and incremental object model to find matches between visual object tracks (Teixeira and Corte-Real, 2008). The description scheme relies on SIFT local descriptors and a text-like bag-of-words representation. Results show that this object representation scheme can be used to aid tracking of generic objects in visual surveillance systems, since it can discriminate a large quantity of different visual objects very well, and can be adapted to reflect object changes. It also presented a good resilience to incorrect segmentations when extracting the visual objects from complex scenes. The object model is updated through incremental learning, avoiding excessive data storage while maintaining performance and allowing new objects to be learnt by the system.

Other analysis algorithms, namely audio content analysis, will be added in the future.

4.2 Prototype

The current prototype of the proposed architecture is based on ZoneMinder [3]. This choice is due to

[3] http://www.zoneminder.com

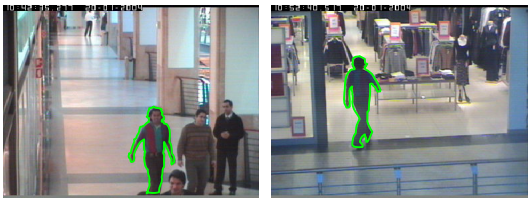

Figure 3: Same visual object captured at different instants, by different cameras.

the fact that ZoneMinder is an open-source surveillance framework that includes functionalities of typical surveillance systems – supports capture, analysis, recording, and monitoring of multiple video cameras. Given a state-of-the-art surveillance framework, which closely follows the architecture defined in Section 3, our goal is to instantiate the requirements stated in Section 2.

The ZoneMinder framework comprises different components written in C++ (core), PHP (GUI) and Perl (scripts). A motion analysis module detects activity in the captured areas, allowing selective recording defined by the user. The user can also define alarms, such as intrusion in protected areas, which are stored in a MySQL database as events. These events are associated with the corresponding video for future visualization. These management and visualization functions are accomplished through a web-based GUI.

The analysis functionalities provided by typical surveillance systems, such as ZoneMinder, are limited to activity detection based on motion analysis and alarm generation based on the definition of protected areas. In the prototype we extended these functionalities with advanced detection such as object segmentation, tracking and matching (see Section 4.1). In summary, and considering the database model proposed in (Black et al., 2004), our prototype populates automatically the semantic layer, in contrast to the typical systems that generate information for the image and motion layers.

5 CONCLUSIONS

This paper introduces a modular surveillance systems architecture supporting the requirements of automated video surveillance networks, which are an increasingly important tool for preventing and countering security threats. The architecture can be adapted to different scenarios. Also, unlike typical surveillance systems, multiple sources of information are considered. A prototype based on open-source framework for visual surveillance was implemented. The

prototype includes algorithms for the segmentation and tracking of visual objects. Another module that detects multiple appearances of the detected visual objects was also devised and integrated in the system. Future work includes integrating of audio analysis modules and the support for evolvability and availability.

REFERENCES

Black, J., Ellis, T., and Makris, D. (2004). A hierarchical database for visual surveillance applications. In *Proceedings of IEEE International Conference on Multimedia and Expo*, pages 1571–1574.

Detmold, H., Dick, A., Falkner, K., Munro, D. S., van den Hengel, A., and Morrison, R. (2006). Scalable surveillance software architecture. In *Proceedings of IEEE International Conference on Video and Signal Based Surveillance*, pages 103–106.

Lee, D.-S. (2005). Effective Gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(5):827–832.

Lo, B., Sun, J., and Velastin, S. (2003). Fusing visual and audio information in a distributed intelligent surveillance system for public transport systems. *Acta Automatica Sinica*, 29(3):393–407.

Pavlidis, I., Morellas, V., Tsiamyrtzis, P., and Harp, S. (2001). Urban surveillance systems: from the laboratory to the commercial world. *Proceedings of the IEEE*, 89(10):1478–1497.

Siebel, N. T. and Maybank, S. J. (2004). The ADVISOR visual surveillance system. In *Proceedings of the ECCV Workshop on Applications of Computer Vision*, pages 103–111.

Teixeira, L. F., Cardoso, J. S., and Corte-Real, L. (2007). Object segmentation using background modelling and cascaded change detection. *Journal of Multimedia*, 2(5):55–64.

Teixeira, L. F. and Corte-Real, L. (2008). Video object matching across multiple independent views using local descriptors and adaptive learning. *Pattern Recognition Letters*. (in press).

Valera, M. and Velastin, S. A. (2005). Intelligent distributed surveillance systems: A review. *IEE Proceedings: Vision, Image and Signal Processing*, 152(2):192–204.

Valera, M. and Velastin, S. A. (2008). Middleware for distributed video surveillance. *IEEE Distributed Systems Online*, 9(2).

Zhao, T. and Nevatia, R. (2004). Tracking multiple humans in complex situations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1208–1221.