A Dynamic Environment for Video Surveillance

Paolo Bottoni[1], Maria De Marsico[1], Stefano Levialdi[1,2], Giovanni Ottieri[1], Mario Pierro[1], and Daniela Quaresima[1]

[1] Department of Computer Science "Sapienza" University of Rome, Via Salaria. 113, 00198 Roma, Italy
[2] DEI, Carlos III University, Madrid
{bottoni,demarsico,levialdi}@di.uniroma1.it, mat101@gmail.com, {jowans,daniela.quaresima}@libero.it

Abstract. Video surveillance systems must support multiple streaming and prompt alert notification. We propose a two-tiered environment: a supervisor defines presentation layouts and model interface reactions to alerts; a surveillant watches synchronized videos, adapts layouts, and is notified with alerts.

Keywords: Video surveillance, Interface reconfiguration, Synchronization.

1 Introduction

Video surveillance is evolving from analog to digital, decoupling video sources from presentation devices and enabling random access to scenes, automatic processing of anomalies and advanced interfaces [1]. However, surveillants, still have to react to alerts and automatic recognition of individuals or suspect behaviours is complex [2,3]: small variations in algorithms parameters cause false negatives / positives, wasting attention resources or diminishing users' trust in the system. An effective interface should support event interpretation via multiple cameras, and ensure proper information on alerts for timely redirection of attention. Hence, we organize subsets of videos into *views*, with suitable stream layouts, typically related to camera arrangement. The interface always presents the whole stream set at reduced resolution; and a scheduler ensures adaptable frame rates for video refresh. Flow synchronization provides optimal resource usage and presentations suited to the task. Stream rearrangement and resynchronization may comply with surveillants' dynamic needs or with supervisor's policies – e.g. to present streams from an alert area. The overall interface structure is kept constant, while content is dynamically adapted adapted to ensure full control whilst reducing attention stress.

2 Related Work and Requirement Analysis

Research on usability of interactive digital surveillance systems is recent. Research prototypes often assume well-controlled and synchronized settings and exclude humans from the processing loop. However, [4] advocates user involvement through

T. Gross et al. (Eds.): INTERACT 2009, Part II, LNCS 5727, pp. 892–895, 2009.

usable interfaces. Focusing on interaction, we consider the impact of usability flaws encountered by surveillants [5], especially related to performance degradation over time and divided attention among cameras. Commercial systems map camera banks onto window banks (*views*): a surveillant selects cameras to trace activities through views. With many cameras, small images make this difficult; moreover, it is hard to anticipate where a traced subject will reappear. Help comes from camera maps, and from experimental layout algorithms arranging peripheral streams according to their relative positions: a "geographic" context helps tracing a subject [6]. Our scheduling derives from Ptolemy's HDF [7]. Each iteration is a state in which flows stay constant and the number of required activation cycles is determined. After each iteration, admissible modifications can result in revision of the scheduling.

Literature and interviews with surveillants provided the main requirements: continuous area overview and rapid focus switch on relevant information in case of alarms. Users also need to modify stream arrangements and frame rates and to relate each stream with the different views. To this aim, our system provides a map and uses identifiers and color codes to indicate camera positions in a window grid in which streams are played. Continuous information is kept for auditing: by logging all modifications of the interface organization, either due to user interaction or to generated alerts, and keeping track of users' accesses. Interviews with experts also highlighted the need for two classes of users, *supervisors* and *surveillants.*

3 System Use and Architecture

A network of services constitutes the server side. One service feeds the interface and the streams, the others perform recognition activities. Fat clients allow access as *Supervisor* or *Surveillant*. Supervisors manage stream identity and layout in a view, and define a set of predefined views to address frequent or critical needs. They also define or add servers for automatic alert triggering, and include or exclude cameras from views, marking them on the map. Supervisors can select sources of different types (files, web-accessible or closed-circuit cameras) and specify their paths and access rights. Both supervisors and surveillants can further define parameters, e.g. source frame rate, position and size. Modifications of frame rates or of the composition of the active view require rescheduling. The surveillant environment implements the *overview + detail* paradigm [8]. A panel constantly provides the whole streaming set in a reduced size and a map of the cameras with positions and observation cones. Such elements are kept constant as top (*Stream*) and bottom-right panels (*Map*). The bottom-left panel (*Focus*) presents the currently selected view: windows presenting streams at the chosen occupy grid cells to allow a closer inspection of specific sources. Fig. 1 shows an interface screen with a chosen *view*. A surveillant can adapt a presentation by adding or removing streams to and from the *Focus*. Modifications cannot be made permanent, except for the dedicated *surveillant view*. Alerts can activate a predefined view, or push specific information in the normally empty *alert cell*. Fig. 1 shows how subject recognition from a database of suspects triggers the visualization of the source stream in the alert cell, while the database record is presented in the *Information Panel* (above the *Map* Panel).

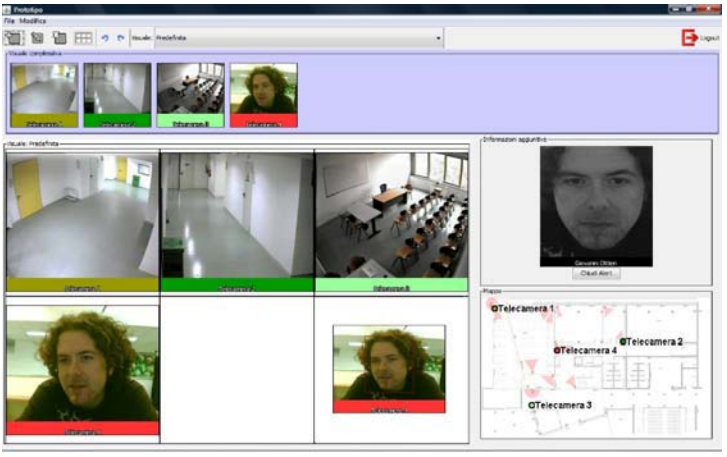

Fig. 1. In an alert situation the *Information* panel presents info on the recognized person, while the live stream in which the person was recognized is shown in a cell of the *Focus* panel

An alert can cause replication of a stream on several windows: potentially dangerous situations are rapidly assessed, by attracting attention through interface changes, possibly complemented by sound signals. Frame rates of streams can be singly adapted, e.g. streams from less visited zones may be kept at a low frame rate, automatically increased if something occurs there. Typically, frame rates in the *Stream* panel are slower than in the *Focus* one. A `Scheduler` manages the single rates. Events are classified according to possible impact on the video presentation rate, and identified by interpreting stimuli from the user or from servers. Each interpretation defines a request, associated with a command sequence, possibly leading to rescheduling and to redefining the *Focus* panel. Each modification request is routed to a specific `Command Manager` for each panel.

4 System Usability

The usability analysis was introduced in the development process through a cognitive walkthrough, and an analysis based on Nielsen's heuristics. We then classified the identified design flaws according to Cognitive Dimensions [9], only considering those relevant to the designed tasks. We especially analyzed view adaptation in the supervisor and surveillant interfaces. As for the notions of *notation*, *environment* and *media* upon which the analysis is based, we respectively identified: a) the collection of windows within the *Focus* panel; b) the user commands to add, remove, or change visualization parameters of selected videos; and c) the *Focus* panel, where the user can manipulate symbols. Due to lack of space we only highlight some examples of the evaluation and redesign process. For Abstraction: types and availability of abstraction mechanisms, we observed that abstraction creation and management are straightforward. However, it was difficult to understand at first sight that the view can contain any view from a menu. A simple workaround was to expand the label of the view

panel. A problem with Premature commitment: constraints on the order of doing things was related to the expected position of a newly added element. We decided to use a fixed grid for positions.

5 Conclusions

We supports surveillants in rapidly focusing attention on relevant sources, while maintaining an area overview. A first tier defines a supervisor environment, while in a second, the surveillant can change predefined settings according to current needs. The system can present relevant streams when predefined events occur. This avoids missing relevant events due to difficult orientation within a stream bank, and reference to physical locations. Users can personalize frame rates.

Acknowledgments. Partially supported by MIUR – PRIN 2006.

References

1. Valera, M., Valestin, S.A.: Intelligent distributed surveillance system: a review. IEEE Proceedings of Image Signal Process 152, 192–204 (2005)
2. Micheloni, C., Snidaro, L., Piciarelli, C., Foresti, G.L.: Exploiting Temporal Statistics for Events Analysis and Understanding. In: Proc. ICIAP 2007, pp. 530–535. IEEE CS, Los Alamitos (2007)
3. Bremond, F., Thonnat, M.: Issues of Representing Context Illustrated by Video-Surveillance Applications. Int. J. of Hum.-Comp. St. 48, 375–391 (1998)
4. Yamasaki, T., Nishioka, Y., Aizawa, K.: Interactive Retrieval for Multi-Camera Surveillance Systems Featuring Spatio-Temporal Summarization. In: MM 2008, pp. 797–800 (2008)
5. Keval, H.U., Sasse, M.A.: Man or a Gorilla? Performance Issues with CCTV Technology in Security Control Rooms. In: 16th World Congress on Ergonomics Conference (2006), http://hornbeam.cs.ucl.ac.uk/hcs/publications/IEA2006_Keval.pdf
6. Girgensohn, A., Shipman, F., Turner, T., Wilcox, L.: Effects of Presenting Geographic Context on Tracking Activity between Cameras. In: Proc. CHI 2007, pp. 1167–1176. ACM Press, New York (2007)
7. Brooks, C., Lee, E.A., Liu, X., Neuendorffer, S., Zhao, Y., Zheng, H.: Heterogeneous Concurrent Model and Design in Java. Ptolemy II Domains, vol. 3, UCB/EECS-2008-37 (2008)
8. Cockburn, A., Karlson, A., Bederson, B.B.: A review of overview+detail, zooming, and focus+context interfaces. ACM Comp. Surv. 41(1), 1–31 (2008)
9. Green, T.R.G., Petre, M.: Usability Analysis of Visual Programming Environments: A 'Cognitive Dimensions' Framework. Journal of Visual Languages and Computing 7, 131–174 (1996)