Exploiting Multimodal Interaction Techniques for Video-Surveillance

Marc Castelló[1], Jordi Gonzàlez[1], Ariel Amato[1], Pau Baiget[1], Carles Fernández[1], Josep Gonfaus[1], Ramón A. Mollineda[2], Marco Pedersoli[1], Nicolás Pérez de la Blanca[3], F. Xavier Roca[1]

Abstract In this paper we present an example of a video surveillance application that exploits Multimodal Interactive (MI) technologies. The main objective of the so-called VID-Hum prototype was to develop a cognitive artificial system for both the detection and description of a particular set of human behaviours arising from real-world events. The main procedure of the prototype described in this chapter entails: (i) *adaptation*, since the system adapts itself to the most common behaviours (qualitative data) inferred from tracking (quantitative data) thus being able to recognize abnormal behaviors; (ii) *feedback*, since an advanced interface based on Natural Language understanding allows end-users the communication with the prototype by means of conceptual sentences; and (iii) *multimodality*, since a virtual avatar has been designed to describe what is happening in the scene, based on those textual interpretations generated by the prototype. Thus, the MI methodology has provided an adequate framework for all these cooperating processes.

1 Introduction

The main objectives of a Video Surveillance (VS) system are typically set to achieve detection of anomalies and/or recognition of a predefined set of agent behaviors arising from real-world events and in real-time [1]. One strategy for embedding *cognition* in VS systems is to convey such recognized events to end-users based on Natural-Language texts. This skill is justified by the fact that we can demonstrate

[1] Centre de Visió per Computador, Dept. Ciències de la Computació, Universitat Autònoma de Barcelona, Barcelona, Spain, e-mail: {mcastello, poal, aamato, pbaiget, perno, gonfaus, marcopede,xavir}@cvc.uab.es
[2] Instituto de Nuevas Tecnologías de la Imagen, Universitat Jaume I, Castelló, Spain, e-mail: ramon.mollineda@lsi.uji.es
[3] Dpto. Ciencias de la Computación e I.A., ETSI Informática y de Telecomunicación, Universidad de Granada, Granada, Spain, e-mail: nicolas@ugr.es

that we understand what is going on in a scene if we are able to describe it using Natural Language [2].

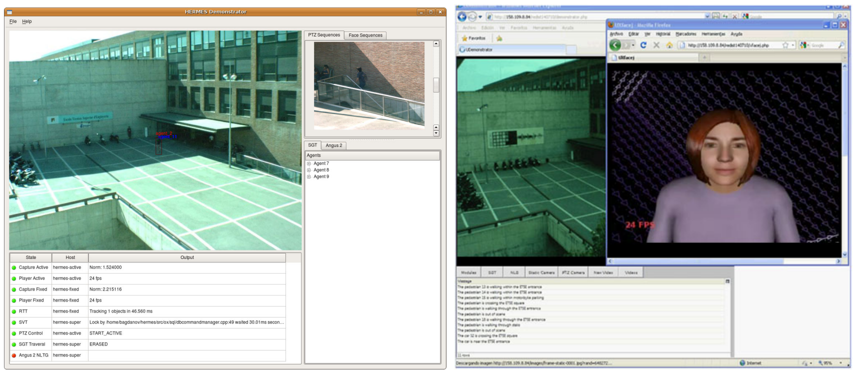
This communication capability is considered an important step towards developing a *Cognitive Video Surveillance* (CVS) system [4]. For example, a CVS system exploiting language capabilities would require to implement the following functionalities [6]: (i) *adaptation*, i.e. the system adapts its recording conditions for best keeping track of agents and for best inferring contextual knowledge; (ii) *feedback*, i.e. an advanced interface based on Natural-Language understanding supports conceptual feedback where input texts define the end-user's queries and commands; and (iii) *multimodality*, i.e. inferred conceptual predicates are converted into Natural-Language sentences for allowing the system its communication with end-users, for example by means of a virtual avatar.

An envisaged prototype focused on the aforementioned cognitive aspects requires the implementation of several modules like, at the very least, to detect motion while keeping track of agents over time; to infer high-level, conceptual descriptions based on agent trajectories combined with context (i.e. interacting objects and knowledge about the scene [3]); and to communicate those inferred interpretations to human operators using Natural-Language texts (and preferably in multiple languages).

This chapter introduces a CVS system as defined before. In essence, the prototype we have developed consists of the following steps:

- Those events detected in image data-streams are obtained from a system composed of static or active cameras, see Fig. 1.(i), for example following the architecture presented in [7].
- Since common background subtraction detection and filter-based tracking allows the system to estimate the trajectories of moving agents within the scene, these trajectories constitute the basis of knowledge for further inference, as in [4].
- Semantic models defined by experts using logic formalism are used to infer interactions and behaviors. The use of ontologies is very helpful for guiding the top-down modelling of the expert database, while providing a framework which centralizes the multiple types of knowledge involved in CVS (e.g. visual, conceptual and linguistic).
- The generation of Natural-Language is usually based on linguistic models which convey those semantic concepts inferred from behavioral models.
- Understanding textual queries from end-users is the basis for feedback, since these queries can modify the functionality of the system by incorporating new knowledge (e.g. refining a description or giving a better estimation of *why* a specific behavior is happening) or by modifying the acquisition conditions.
- Lastly, an advanced, multimodal interface is designed to communicate the results to end-users using texts [5]. These Natural-Language sentences are reported by means of a virtual avatar who *explains* us using speech what is happening in the scene, see Fig. 1.(ii).

In the sequel, we will describe the methodology used to implement the capabilities of adaptation, feedback and multimodality in a surveillance prototype.



State	Host	Output
● Capture Active	hermes-active	Norm: 1.524000
● Player Active	hermes-active	24 fps
● Capture Fixed	hermes-fixed	Norm: 2.215116
● Player Fixed	hermes-fixed	24 fps
● RTT	hermes-fixed	Tracking 1 objects in 46.560 ms
● SVT	hermes-super	Lock by /home/bagdanov/hermes/src/ox/sql/dbcommandmanager.cpp:49 waited 30.01ms secon...
● PTZ Control	hermes-active	START_ACTIVE
● SGT Traveral	hermes-super	ERASED
● Angus 2 NLTG	hermes-super	

```
Message
The pedestrian 13 is walking within the ETSE entrance
The pedestrian 14 is walking within the ETSE entrance
The pedestrian 15 is walking within motorbyke parking
The pedestrian is crossing the ETSE square
The pedestrian is walking through the ETSE entrance
The pedestrian is out of scene
The pedestrian 18 is walking through the ETSE entrance
The pedestrian is walking through stairs
The pedestrian is out of scene
The car 32 is crossing the ETSE square
The car is near the ETSE entrance
11 rows
Descargando imagen http://158.109.8.84/images/frame-static-0001.jpg?rand=648272...
```

(i) (ii)

Fig. 1 Examples of advanced interfaces exploiting Natural-Language, which support conceptual feedback plus a communication with human operators using virtual avatars.

2 Methodology and theoretical background

2.1 Adaptation: Anomaly Detection based on Trajectory Analysis

Nowadays, the detection of anomalies in video sequences is considered a hot topic in video understanding research [6]. This issue is caused not by the difficulty of implementing an anomaly detector, but because it is unclear which is the best definition of anomaly. On the one hand, the concept of anomaly is usually related in video-surveillance to suspicious or dangerous behaviors, i.e. those for which an alarm should be fired when detected.

From a statistic point of view, normal behavior occurs more frequently than anomalous behavior. This is the main assumption of all works performed in anomaly detection research [12, 11, 4]. Since an anomaly is a deviation from what is considered normal, these two concepts, normality and anomaly, are complementary. In the video-surveillance domain, standard learning procedures use observations extracted by a motion tracking algorithm over a continuous recording to build a model of the scenario that will determine somehow the normality or abnormality of new observations: trajectories, understood as the series of positions of an object over time, from entering to exiting a scene, are considered by most authors as the most useful information to embed the behavior of moving objects [15].

Extensive work has been done on behavior understanding based on trajectory analysis: Makris and Ellis [4] considered spatial extensions of trajectories to construct path models, which were updated when new trajectories were matched. A similar approach was previously used in [14]. Piciarelli and Foresti [9] presented an online modeling algorithm to obtain a hierarchy of typical paths. Hu et al. [11] obtained motion patterns by spatially and temporally clustering trajectories using fuzzy c-means. More Recently, Basharat et al. [12] modeled a probability density

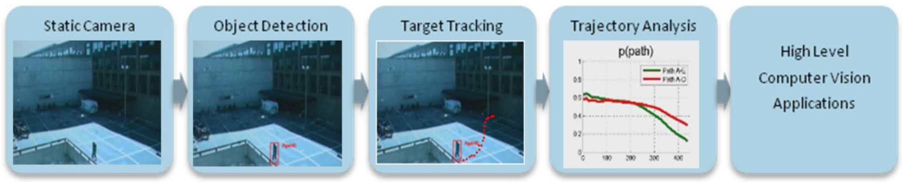


Fig. 2 Main steps for anomaly detection.

function at every pixel location by means of Gaussian Mixture Models (GMM), considering not only spatial coordinates but also object sizes.

Inspired in these works, the procedure depicted in Fig. 2 is applied for each pair $(s,e) \in SxE$. Let $T_{s,e} = \{t_1,\dots,t_N\}$ be the set of trajectories starting at position s and ending at e. Each trajectory t_i is represented by a sequence $r(t_i)$ of K equally spaced control points sampled from the tracked points $P_{s,e} = \{p_{i_1},\dots,p_{i_K}\}$

The input of the algorithm consists of the set of trajectories starting in an entry point $s \in S$ and ending in an exit point $e \in E$. These trajectories are represented in a $K \times N$ matrix, where N is the number of trajectories and K is the number of control points that have been sampled for each trajectory. Each trajectory is associated to one of the routes R_c that form the trajectory $P_{s,e}$. This is represented by points (k,c), which contains the list of k-th points of all the trajectories being currently associated to the route c.

Subsequently, the number of Gaussian components that best represent each route, see 3 route examples in Fig. 3. The list of trajectories for each route is modeled using a one-component and a two-component GMMs. After applying the algorithm to each pair of entry and exit areas, M contains the set of normal paths that trajectories should pass in the future. Thus, a trajectory deviating from M will be considered as an anomaly.

Thus the prototype is able to differentiate among three kinds of anomaly with respect to the current trajectory over the scenario, see Fig. 4:

- Soft Anomaly (SA): Some parts of a trajectory t_a are classified as SA if they follow a modeled path, but there are sudden changes in speed or orientation that differ from the learnt routes from s to e.
- Intermediate Anomaly (IA): A trajectory t_b is classified as an IA if the most probable path $P_{s,e}$ changes from one learnt route to another.
- Hard Anomaly (HA): A trajectory t_c is classified as a HA if it has performed a completely unobserved path. This can be caused because the trajectory started from an entry point e/E or because the probability of any path beginning in the actual entry point is too low for the whole scene.

This description of anomalies represents different degrees of deviation between a new observation and the learnt model. Indeed, SA can be considered as a route deviation inside a path. IA detects path deviations inside the same model M. Finally, HA represents a complete deviation from M.

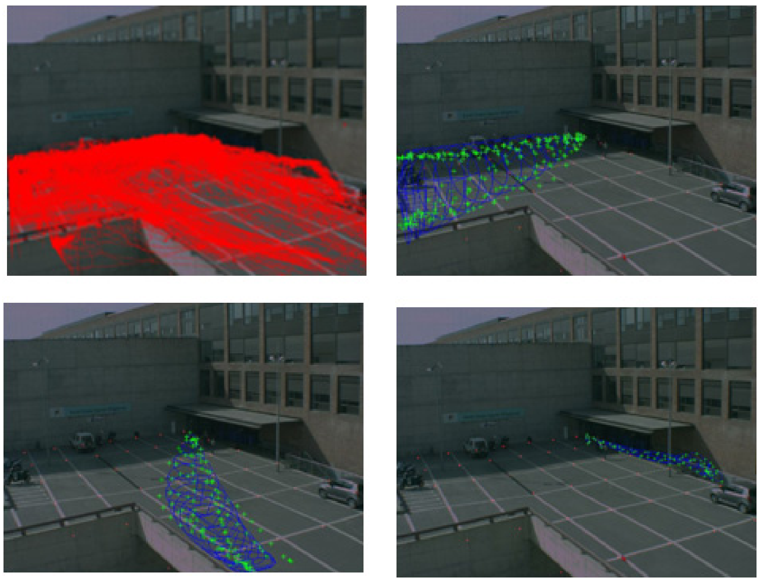


Fig. 3 Learned trajectories and the probabilistic paths of three routes.

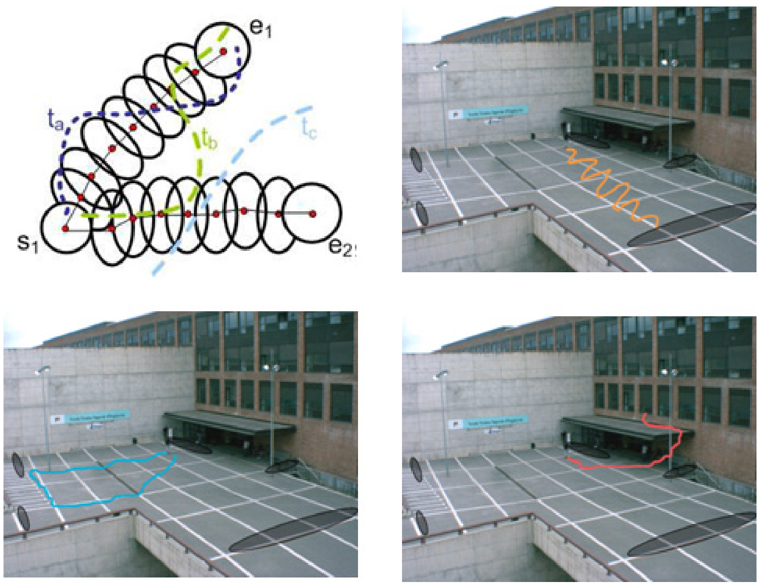
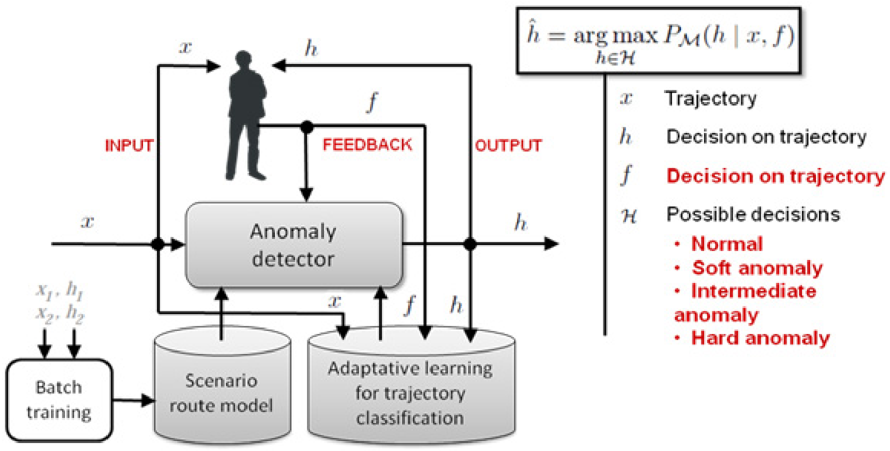


Fig. 4 Examples of the different types of anomalies detected in our prototype.

Summarizing, Fig. 5 shows the modular schema of our anomaly detector within the Multimodal Interactive paradigm developed during the MIPRCV project.



$\hat{h} = \arg\max_{h \in \mathcal{H}} P_{\mathcal{M}}(h \mid x, f)$
x Trajectory
h Decision on trajectory
f Decision on trajectory
$\mathcal{H}$ Possible decisions
• Normal
• Soft anomaly
• Intermediate anomaly
• Hard anomaly
INPUT
FEEDBACK
OUTPUT
Anomaly detector
Batch training
Scenario route model
Adaptative learning for trajectory classification
x_1, h_1
x_2, h_2

Fig. 5 Adaptive model for anomaly detection.

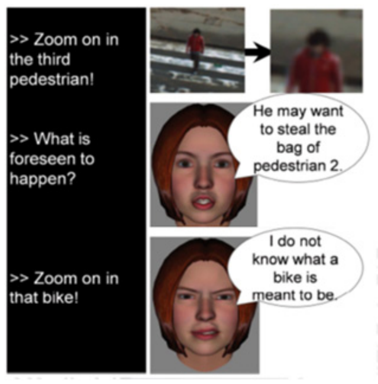
2.2 *Feedback: Interaction based on Natural-Language Generation and Understanding*

Natural Language Generation (NLG) and Natural Language Understanding (NLU) are both subfields of Natural Language Processing (NLP), which in turn can be seen as subfields of both computer science and cognitive science [23, 6]. NLG focuses on computer systems which can automatically produce understandable texts in a natural human language, and NLU studies computer systems which understand these languages, see Fig. 6.(i). Both NLG and NLU are concerned with computational models of language and its use.

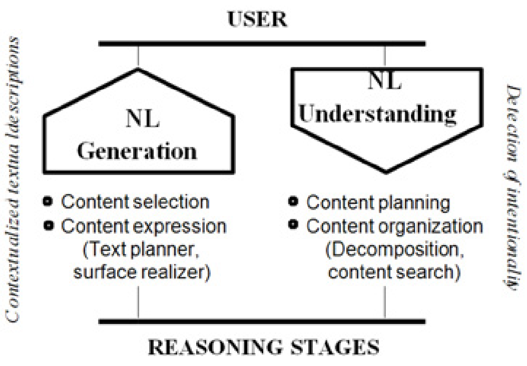
In general terms, the two processes have the same end points but opposite directions, see Fig. 6.(ii).Nevertheless, the internal operations of these processes hold several differences in character. NLG has been often considered as a process of choice, whereas, NLU has been best characterized as one of hypothesis management. In NLG, we have several means available, and must choose the most suitable one to achieve some desired end. In NLU, we must select the most appropriate interpretation out of a multiple set of them, given some input.

Therefore, the two strategies which have been considered to design the NL interface are different for the two NLG vs. NLU processes. In NLG we control the set of situations which need to be expressed, and can define one correct form of expressing that information in a clear and natural way for each language considered. On the other hand, the NLU process provides us with an open number of possible user queries which need to be interpreted; we need to somehow restrict to a set of intentions which we assume the user can show.

From a general perspective, some characteristics have been considered for NLG:


>> Zoom on in the third pedestrian!
>> What is foreseen to happen?
He may want to steal the bag of pedestrian 2.
>> Zoom on in that bike!
I do not know what a bike is meant to be.

(i)


USER
NL Generation
NL Understanding
Content selection
Content expression (Text planner, surface realizer)
Content planning
Content organization (Decomposition, content search)
REASONING STAGES
Contextualized textual descriptions
Detection of intentionality

(ii)

Fig. 6 (i) Examples of interfaces exploiting Natural-Language and (ii) NLG vs. NLU processes.

- We describe situations contained in the implemented behavioral model. In our case, the situations are those ones defined in a domain ontology and used for behavioral analysis in Situation Graph Trees [25].
- Following the cognitive situatedness/embeddedness property, outputs have been restricted to interpretations of the possible situations uniquely for the defined domain [24]. These interpretations will be expressed linguistically by native speakers of each language.
- Such linguistic utterances are built and adapted into the system using rule based parsing techniques and functional grammars [20].
- The linguistic model is based on the Prototype Theory from cognitive linguistics, in which elements are categorized using sets of semantic features [22]. As explained in some of the following chapters, this approach entails a series of advantages, like the lack of rigidness to formalize linguistic properties, or the interoperability of linguistic knowledge at different stages.

There are several operations fulfilled by the NLG module to generate textual descriptions in NL from high-level semantic predicates [19, 21]. Basically, this module uses two kinds of knowledge sources, ontological/situational and linguistic; three kinds of grammar, ranging from semantic to morphological considerations; and an onomasticon to handle the history of instantiations during a discourse.

On the other hand, NLU has usually been regarded as a process of hypothesis management that decides for the most probable interpretation of a linguistic input. Following this idea, the NLU module links plotline sentences to their most accurate interpretations in the domain of interest, in form of high-level predicates referring to known concepts or instances within the scene. In essence, we detect all valid queries which apply to predefined goals in the domain of interest, which can be generally classified as questions, commands, or information updates. An ontology is used to restrict the semantic domain of validity of these queries.

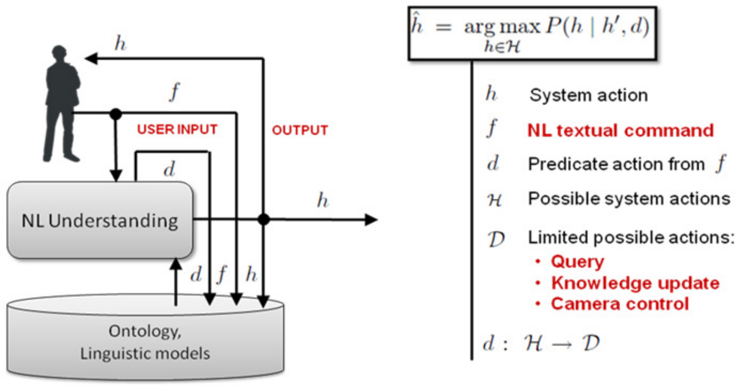

$\hat{h} = \arg\max_{h \in \mathcal{H}} P(h \mid h', d)$
h
f
USER INPUT
OUTPUT
d
NL Understanding
h
d f h
Ontology, Linguistic models
h System action
f NL textual command
d Predicate action from f
$\mathcal{H}$ Possible system actions
$\mathcal{D}$ Limited possible actions:
• Query
• Knowledge update
• Camera control
$d : \mathcal{H} \to \mathcal{D}$

Fig. 7 Feedback model for Natural Language interaction.

Once a proper formatting has been applied, an input sentence is analyzed through a sequence of 3 processes: first, a morphological parser tags each input word with linguistic features depending on the context of apparition. Secondly, a syntactic/semantic parser recursively builds a dependency tree for the tagged sentence. Finally, the resulting dependency tree, already having ontological references, is assigned to the most related high-level predicate found [25]. The morphological and syntactical processes convert the NL sentence into a tree representation, which is finally converted into a goal predicate measuring the distances from the tree to the predicates stored in the ontology. Subsequently, each plotline predicate produced by the NLU module instantiates a high-level event, which must be converted into a list of explicit spatiotemporal actions which are conveyed to the end-user.

Summarizing, Fig. 7 shows the modular schema of our Natural Language interaction within the MI paradigm developed during the MIPRCV project.

2.3 *Multimodality: Animation of Virtual Avatars for Communication with End-Users*

XfacePlayer is an application designed by the Cognitive and Communication Technologies (TCC) division of ITC-irst and modified for the animation of 3D talking heads through either the MPEG-4 Facial Animation standard or key frame interpolation[1]. XfacePlayer relies on Microsoft's Speech API 5.1 Text-To-Speech engine and NeoSpeech's Kate16 voice for speech synthesis in order to generate the voice for the model. The version used in the VID-Hum prototype is written in java and works in a web browser, see Fig. 8.

[1] http://xface.fbk.eu/index.htm

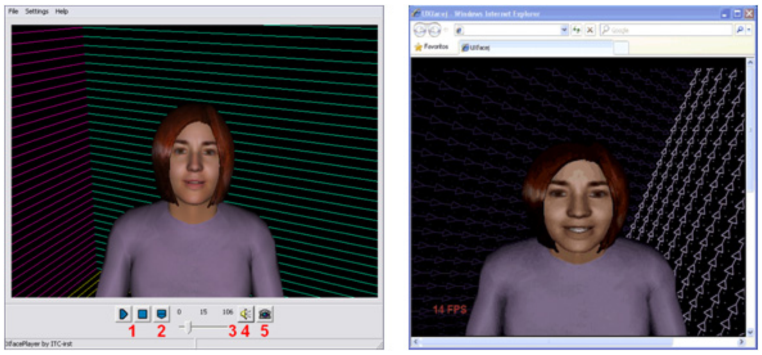


Fig. 8 Multimodal communication using virtual avatars: XfacePlayer interface adapted to the xMontse model(left), and its web version (right).

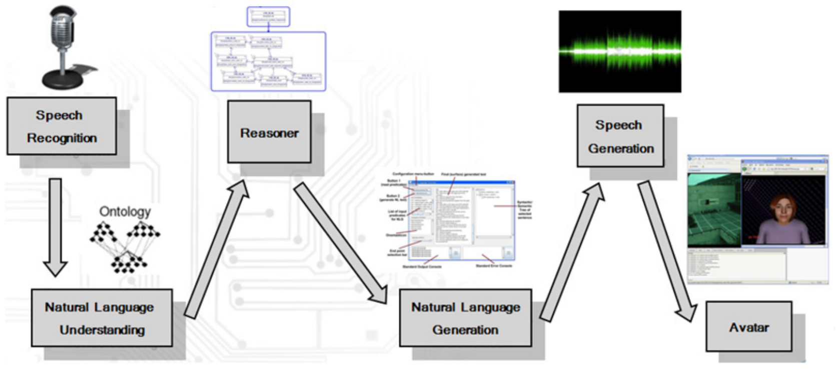


Fig. 9 Multimodal interaction complete process from voice commands or queries to the response using an avatar and a speech generation module.

From the multimodal point of view, the system depicted in Fig. 9 is a complete set of modules that goes from recognizing the human speech to reply simulating an intelligent virtual human answer and/or doing some action. Voice recognition module implements a signal (voice) to text transformation. Speech synthesis (TTS), along with the avatar animation, implements a text-to-signal transformation. The reasoner is designed to fit in the middle, to provide the text-to-text transformation that appears to produce an intelligent reply to the human input, and could execute some instructions like to zoom in a pedestrian. The Avatar allows any user to see a virtual head pronouncing any given text while displaying facial gestures and emotions.

Summarizing, Fig. 10 shows the modular schema of our multimodal model within the MI paradigm developed during the MIPRCV project.

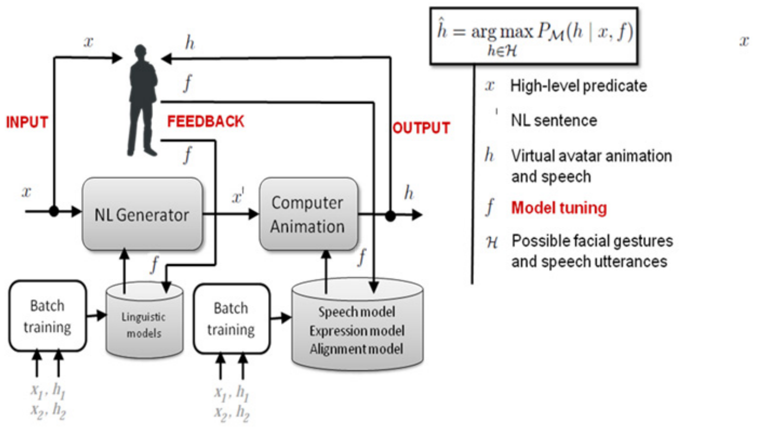


Fig. 10 Multimodal model for human-computer communication.

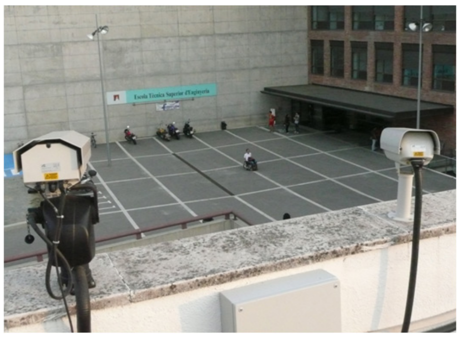


Fig. 11 The scene is surveyed from atop the CVC building using one fixed (right one in the image) and one active camera (left).

3 The VID-HUM demonstrator

The prototype platform has been installed on top of the CVC building in Barcelona. The cameras survey a scene in front of the engineering building on the campus of the Universitat Autònoma de Barcelona, see Fig. 11 for an example scene view. The hardware platform for the VID-Hum prototype consists of one fixed camera and a camera mounted in a pan/tilt platform and fitted with a zoom lens.

The hardware platform for the prototype consists of one fixed camera, a camera mounted in a pan/tilt platform and fitted with a zoom lens, three dedicated servers to provide raw computational power and a fast 1 Gb Ethernet switch. The hardware integration architecture is illustrated in Fig. 12. The main components of the hardware infrastructure are Cameras, PTZ platform, Compute Servers and Network Infrastructure, as described next.

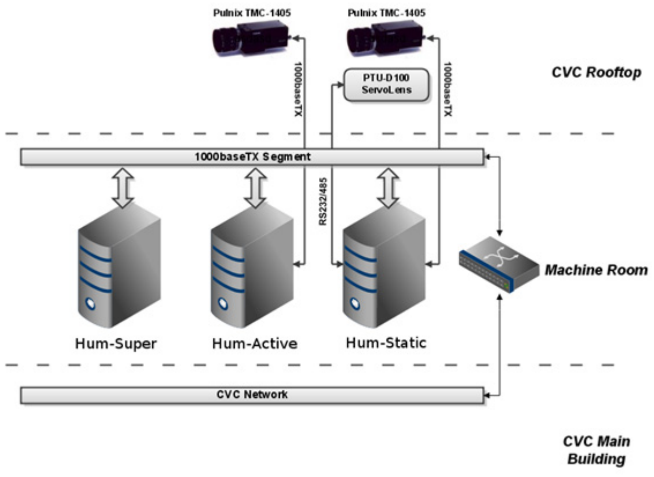


Fig. 12 The VID-Hum prototype hardware infrastructure.

Two Pulnix TMC-1405GE cameras are used for the VID-Hum prototype. These cameras are GigE-compatible and deliver very high-resolution (1392x1040) imagery at high framerate (30fps). Each camera is connected by a dedicated, 1000base-TX Ethernet connection to ensure constant, high-framerate streaming from both cameras. The connection between Servolens zoom optics and the active pc is done by an RS-232 serial communication, at 9600 baud of speed. One of the Pulnix cameras is mounted in a Directed Perception PTU-D100 pan/tilt platform that allows 350° pan and 180° tilt surveillance. It also allows a wide range of pan and tilt speeds (0.0075°/sec to over 120°/sec), and is fully sealed for outdoor operations.

Three dedicated Dell Poweredge T100 servers are used. Two of these are directly connected to the Pulnix cameras and are primarily dedicated to video acquisition. The third server is used for components not requiring direct access to the cameras, such as the supervisor tracker and SGT reasoning subsystems described below. These three machines are referred to as Hum-Super, Hum-Static, and Hum-Active to emphasize their roles in the prototype platform. Every computer has 3,2GB of RAM of up to 8GB possible if was necessary, one Intel Dual Core Xeon E3120 at 3.16GHz with two processors, 300GB of Hard Disk and Ubuntu 8,10 (Intrepid) OS with kernel 2,6,27,11-generic.

Fig. 13 shows the modular software architecture designed for the VID-Hum prototype platform. There are two components in the prototype architecture that greatly aided integration: the MySQL database server used to communicate between medium-to-high-level components and the use of the Player/Stage system to provide a layer of hardware abstraction between the VID-Hum prototype and the low-level hardware of the cameras and the PTZ platform.

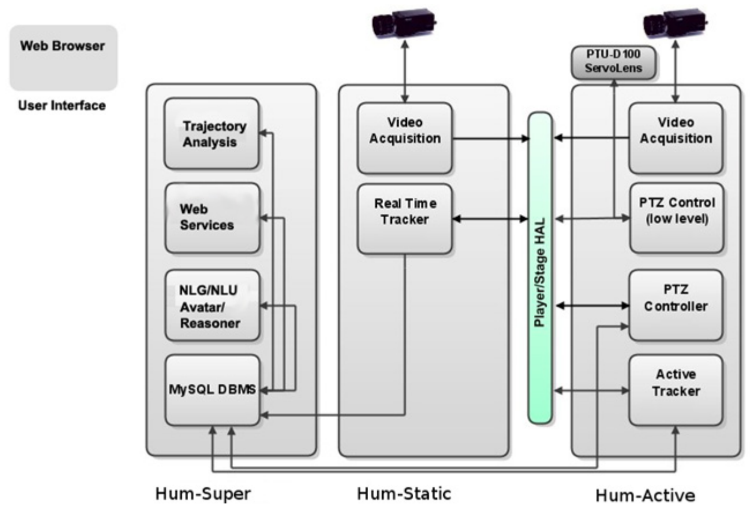


Fig. 13 The VID-Hum prototype software architecture.

The major software components in the prototype architecture are: Video Acquisition, PTZ Controller (low level), Real Time Tracker, Avatar, Trajectory Analysis, PTZ Controller, The Reasoner, NLTG, NLTU Player/Stage Hardware Abstraction Layer, and MySQL DBMS. A Real Time Tracker (RTT) is one of the fundamental components in the prototype platform. The tracker software communicates through a standardized interface, which allows the parallel development of the prototype as well as the tracking method itself. The RTT tracks multiple moving targets in the fixed camera view and writes its observations into a table on the MySQL server.

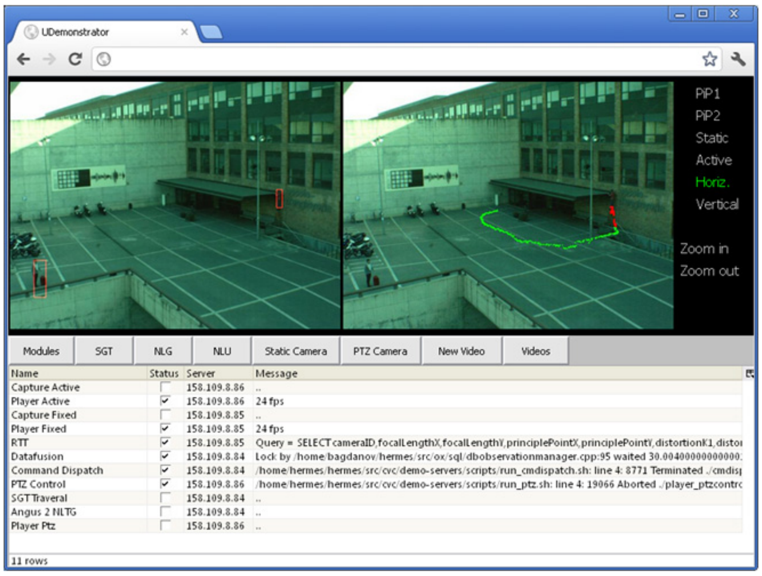
The main features and components of the prototype interface are:

Administration and Monitoring of Components: Given the number and diversity of components in the prototype, administration and monitoring of these distributed components is one of the main functions of the user interface. The interface also allows individual components or subsets of components to be tested in isolation from the others.

Visualization of Video Streams: The largest panel in the graphical interface, in the upper left of Figure 14, displays video streaming from one or both prototype cameras. Streaming video can be viewed from each camera individually, or simultaneously in split-screen mode.

Visualization of Tracker Telemetry: When the tracker is active, the observations emanating from it are displayed in real-time, overlayed on the streaming video from the fixed camera.

Trajectory Analysis: The module tracks people and then studies if the trajectories are normal or not, without any deterministic information of the scene, the trajectory is classified and its result is showed in real time as an overlay to the camera stream.



Name	Status	Server	Message
Capture Active	☐	158.109.8.86	..
Player Active	✓	158.109.8.86	24 fps
Capture Fixed	☐	158.109.8.85	..
Player Fixed	✓	158.109.8.85	24 fps
RTT	✓	158.109.8.85	Query = SELECT cameraID,focalLengthX,focalLengthY,principlePointX,principlePointY,distortionK1,disto
Datafusion	✓	158.109.8.84	Lock by /home/bagdanov/hermes/src/ox/sql/dbobservationmanager.cpp:95 waited 30.00400000000000
Command Dispatch	✓	158.109.8.84	/home/hermes/hermes/src/cvc/demo-servers/scripts/run_cmdispatch.sh: line 4: 8771 Terminated ./cmdis
PTZ Control	✓	158.109.8.86	/home/hermes/hermes/src/cvc/demo-servers/scripts/run_ptz.sh: line 4: 19066 Aborted ./player_ptzcontr
SGT Traveral	☐	158.109.8.84	..
Angus 2 NLTG	☐	158.109.8.84	..
Player Ptz	☐	158.109.8.86	..

11 rows

Fig. 14 The GUI for the real-time active surveillance prototype.

Visualization of Reasoner inferences: When the reasoner component is active, inferences coming from it are displayed in the SGT tab. The time-stamp and predicate from each inference are displayed in a list.

Visualization of NLG: The NLG tab allows to see the texts generated from Natural Language Text Generation.

Command window for NLU: The NLU tab allows to enter commands or queries in natural language and see its results.

Virtual Avatar: The Avatar allows any user to see a virtual head pronouncing any given text while displaying emotions and other facial gestures in a 3D environment. When the avatar is selected a new window appears and it tells us what the natural language module is generating.

Summarizing, the VID-Hum prototype platform consists of many distributed software components. So a web graphical user interface was designed and implemented to organize, administer and visualize the operation of these diverse components. Fig. 14 illustrates the VID-Hum prototype GUI. The interface allows visualization of both camera streams individually or in split-screen mode which simultaneously streams scaled versions of both camera streams.

4 Discussion

In this chapter we have detailed a proper architecture for a multimodal video surveillance application. Although the results are presented using a single outdoor scenario, the modular design of the architecture makes our prototype suitable to be installed in other scenarios. That means, the low-level tasks should be adapted depending on the characteristics of the new scene, like the motion detection and tracking procedures plus the design of a 3D calibrated, conceptual scene model. For example, in our current implementation, the tracking system is unable to cope well with crowds and multiple occlusions: therefore, the tracking strategy should be substituted if applied to scenarios with more complex pedestrian behaviours. On the other hand, the 3 types of anomalies presented in this chapter are statistically learnt from tracked routes, so the basic procedure for detecting these anomalies would adapt well to other scenes if tracking works well: in our particular implementation, the quantitative results and details of the tracking system using in VID-Hum can be found in [20].

The proof-of-concept in this chapter is that Natural-Language and Virtual Avatars are appropriate when incorporating multimodality and feedback in surveillance prototypes. So, regarding the high-level tasks of the prototype (NLU and NLG, for example), these modules are independent from the scene: in these cases, since a MySQL database server is used to communicate between tracking-to-understanding components, modularity is granted. In addition to a proper quantitative evaluation of the NL modules implemented in this chapter (which can be found in [5] showing how accurate and useful a NL framework can be in surveillance scenarios), we have shown that there are several advantages when converting numerical data obtained from tracking to the conceptual data used by NL: dealing with abstract symbols, generating complex queries, assisting low-level motion tasks in a top-down fashion, describing high-level events, reasoning about the logic terms instantiated from raw pixels, predicting plausible future events, etc. Basically, the use of NL makes possible that the final output of the system is a text which not only describes, in human-readable terms, what is happening but also explains the interactions which take place while inferring why a particular behaviour is being detected.

Lastly, the use of virtual avatars (merging visual and acoustic features) for surveillance purposes has several advantages, as demonstrated in [25]: we consider the use of virtual avatars as an intelligent multimodal affective interface since it combines speaking features like prosody, gaze and spoken words together with visual cues like head nods and lips motion. The aim in the VID-Hum prototype is not only to perceive the virtual avatar as an intentional agent, but also in the future as a means to evaluate the virtual agents behaviours based on the knowledge (and experience) of the human agent.

5 Conclusions

The ability to communicate is innate in a natural cognitive system. There exist several ways to reach this goal artificially, although Natural Language is usually taken as a primary choice, being a flexible, unconstrained, and economical tool that is also intrinsic to end-users. In the MIPRCV project, we have developed linguistic modules to close the communication loop between the system and external users within the surveillance domain.

The main procedure of the prototype described in this chapter entails: (i) *adaptation*, since the system adapts itself to the most common behaviours (qualitative data) inferred from tracking (quantitative data) thus being able to recognize abnormal behaviors; (ii) *feedback*, since an advanced interface based on Natural Language understanding allows end-users the communication with the prototype by means of conceptual sentences; and (iii) *multimodality*, since a virtual avatar has been designed to describe, using synthetic speech, what is happening in the scene, based on those textual interpretations generated by the prototype. Summarizing, MI provided an adequate framework for all these cooperating processes.

In the implementation of the prototype, instrumental to the success of the integration effort was the adoption of a MySQL interface for communication between nearly all components of the prototype platform. This, along with the adoption of the Player/Stage hardware abstraction layer for routing video and PTZ commands, greatly eased integration efforts. In addition to easing integration, the database of observations preserves a complete picture of everything that happens in the scene of surveillance, including inferences and PTZ camera actions. This, combined with recorded video, allows detailed after-the-fact inspection of the results of cooperative detection and recognition of human actions.

Acknowledgements

The authors acknowledge the support of the Spanish Research Programs Consolider-Ingenio 2010: MIPRCV CSD200700018; Avanza I+D ViCoMo (TSI-020400-2009-133) and DiCoMa (TSI-020400-2011-55); along with the Spanish projects TIN2009-14501-C02-01 and TIN2009-14501-C02-02.

References

1. Fusier F., Valentin V., Brémond F., Thonnat M., Borg M., Thirde D., Ferryman J.: Video understanding for complex activity recognition. Machine Vision and Applications, vol. 18, n. 3, pp. 167-188 (2007).
2. Arens M., and Gerber R., Nagel H.–H.: Conceptual representations between video signals and natural language descriptions. Image and Vision Computing, vol. 26, n. 1, pp. 53-66 (2008).

3. Dee H., Fraile R., Hogg D., Cohn A.: Modelling scenes using the activity within them. In: Spatial Cognition VI. Learning, Reasoning, and Talking about Space, pp. 394-408 (2008).
4. Makris D., Ellis T., Black J.: Intelligent Visual Surveillance: Towards Cognitive Vision Systems. The Open Cybernetics and Systemics Journal, vol. 2, pp. 219-229 (2008).
5. Fernández C., Baiget P., Roca F.X., Gonzàlez, J.: Determining the Best Suited Semantic Events for Cognitive Surveillance. Expert Systems with Applications, vol. 38, n. 4, pp. 4068-4079 (2011).
6. Gonzàlez J., Rowe D., Varona J., Roca F.X.: Understanding Dynamic Scenes based on Human Sequence Evaluation. Image and Vision Computing, vol. 27, n. 10, pp. 1433-1444 (2009).
7. Bellotto N., Sommerlade E., Benfold B., Bibby C., Reid I., Roth D., Van Gool L., Fernández C., Gonzàlez J.: A Distributed Camera System for Multi-Resolution Surveillance. In: 3rd ACM/IEEE International Conference on Distributed Smart Cameras (2009).
8. Makris D., Ellis T.: Learning semantic scene models from observing activity in visual surveillance. IEEE Trans. on Systems Man and Cybernetics-Part B, vol. 35, n. 3, pp. 397-408 (2005).
9. Piciarelli C., Foresti G. L.: On-line trajectory clustering for anomalous events detection. Pattern Recognition Letters, vol. 27, n. 15, pp. 1835-1842 (2006).
10. Johnson N., Hogg D.C.: Learning the distribution of object trajectories for event recognition. In: British Machine Vision Conference, pp. 583-592 (1995).
11. Hu W., Xiao X., Fu Z., Xie D.: A system for learning statistical motion patterns. IEEE Trans. on PAMI, vol. 28, n. 9, pp. 1450-1464 (2006).
12. Basharat A., Gritai A., Shah M.: Learning object motion patterns for anomaly detection and im-proved object detection. In: IEEE Conference on CVPR (2008).
13. Hu W., Xie D., Tan T.: A Hierarchical Self-Organizing Approach for Learning the Patterns of Motion Trajectories. IEEE Trans. on Neural Networks, vol. 15, n. 1, pp. 135-144 (2004).
14. McKenna S., Nait-Charif H.: Summarizing Contextual Activity and Detecting Unusual Inactivity in Supportive Home Environment. Pattern Analysis and Applications Journal, vol. 7, n. 4, pp. 386-401 (2004).
15. Zhang Z., Huang K., Tan T., Wang L.: Trajectory Series Analysis based Event Rule Induction for Visual Surveillance. In: IEEE Conference on CVPR (2007).
16. Yao B., Wang L., Zhu S.: Learning a Scene Contextual Model for Tracking and Abnormality Detection. In: IEEE Conference on CVPR Workshops 2008.
17. Morris B., Trivedi M.: Learning trajectory patterns by clustering: Experimental studies and compara-tive evaluation. In: IEEE Conference on CVPR (2009).
18. Bremond F., Thonnat M., Zuniga M.: Video understanding framework for automatic behavior recognition. Behavior Research Methods, vol. 3, n. 38, pp. 416-426 (2006).
19. Arens M., Nagel H.-H.: Behavioral knowledge representation for the understanding and creation of video sequences. In: 26th German Conference on Artificial Intelligence (KI-2003), pp. 149-163 (2003).
20. Fernández C., Baiget P., Roca F.X., Gonzàlez J.: Interpretation of Complex Situations in a Semantic-Based Surveillance Framework. Signal Processing: Image Communication, vol. 23, n. 7, pp. 554-569 (2008).
21. Gerber R., Nagel H.-H.: (Mis-?)Using DRT for Generation of Natural Language Text from Image Sequences. In: IEEE Conference on ECCV, pp. 255-270 (1998).
22. Lakoff G.: Women, fire, and dangerous things. In: University of Chicago Press (1987).
23. Reiter E., Dale R.: Building Natural Language Generation Systems. In: Cambridge University Press (2000).
24. Wilson R.A., Keil F.C. (editors): The MIT Encyclopedia of the Cognitive Sciences. Bradford Books (2001).
25. Fernández C., Baiget P., Roca F.X., Gonzàlez J.: Augmenting video surveillance footage with virtual agents for incremental event evaluation. Pattern Recognition Letters, vol. 32, n. 6, pp. 878-889 (2011).