Journal of Universal Computer Science, vol. 21, no. 6 (2015), 856-870
submitted: 20/10/14, accepted: 20/2/15, appeared: 1/6/15 © J.UCS

Video Semantic Analysis Framework based on Run-time Production Rules - Towards Cognitive Vision

Alejandro Zambrano
(Universidad de la Sabana, Bogotá, Colombia
alejandrozakr@unisabana.edu.co)

Carlos Toro
(Vicomtech-IK4, San Sebastián, Spain
ctoro@vicomtech.org)

Marcos Nieto
(Vicomtech-IK4, San Sebastián, Spain
mnieto@vicomtech.org)

Ricardo Sotaquirá
(Universidad de la Sabana, Bogotá, Colombia
ricardo.sotaquira@unisabana.edu.co)

Cesar Sanín
(University of Newcastle, Newcastle, Australia
cesar.maldonadosanin@newcastle.edu.au)

Edward Szczerbicki
(Gdansk University of Technology, Poland
Edward.Szczerbicki@newcastle.edu.au)

Abstract: This paper proposes a service-oriented architecture for video analysis which separates object detection from event recognition. Our aim is to introduce new tools to be considered in the pathway towards Cognitive Vision as a support for classical Computer Vision techniques that have been broadly used by the scientific community. In the article, we particularly focus in solving some of the reported scalability issues found in current Computer Vision approaches by introducing an experience based approximation based on the Set of Experience Knowledge Structure (SOEKS). In our proposal, object detection takes place client-side, while event recognition takes place server-side. In order to implement our approach, we introduce a novel architecture that aims at recognizing events defined by a user using production rules (a part of the SOEKS model) and the detections made by the client using their own algorithms for visual recognition. In order to test our methodology, we present a case study, showing the scalability enhancements provided.

Keywords: video surveillance, video event recognition, video analysis
Categories: D.2.11, D.2.13, I.2.10, I.4.8, J.7, K.6.5, L.1.3, L.1.4, M.1, M.8

1 Introduction

Event identification in video footage has many important implications for society and industry at large ([Thonnat, 2008], [Zheng, 2014], [Jung, 2011]). In a security and surveillance scenario, it is mandatory to be able to quickly respond to suspicious events (burglary, strange objects, people masses in movement, etc.). Most of these activities require an automated recognition of such events [Aggarwal, 2011]. The classical approach to video annotation is to devote a person to the content analysis of video streams. The aforementioned approach undoubtedly causes many problematic issues like: (i) operator's fatigue, (ii) decreased attention span, and (iii) efficiency loss. It is in this scenario, where computer supported or automated analysis of video emerges as a need. Since large amounts of data need to be processed, algorithmic approaches coming from classical computer vision can be used for segmenting relevant parts of a video in order to identify interesting situations. However, a number of difficulties have been reported with current methods ([Thonnat, 2008], [Ling, 2012], [Jung, 2012]). For example, it is common in current approaches that most of the code when developing a new application (even with similar algorithms) has to be rewritten when the conditions, situation or client specifications varies [Little, 2013]. The aforementioned situation occurs because the focus on video segmentation enhancement arguably leaves the final application to be implemented via switch-cases, which is difficult to maintain. As the system grows, adding a new consideration (not originally covered) becomes a monolithic situation. The abovementioned situation evolves in development slowdowns and implementations, which reuse only a small portion of code, hereafter losing time and money. In this paper, we attempt to mitigate this problem by presenting a flexible architecture, which allows the separation of the logic layer from the computer vision algorithmic containers. In this way, we present such logic as part of a consumable service that will require simple production rules. In order to keep the flexibility of our service, said production rules could be changed at runtime. Our system has proved effectiveness on the detection of semantic events in videos, adapting easily to different requirements by changing the rule sets to be used.

This paper is arranged as follows: in section two, we present some relevant related work; in section three, we introduce our proposed architecture with an example of a production rule; in section four, we present a case example with different scenarios and an evaluation comparing a manually annotated video stream with our results; and lastly, in section five, we present future work and conclusions.

2 Related Work

Creating a cognitive system with visual capacity requires skills both in cognitive systems and image processing for object recognition [Maillot, 2004]. The literature reports some attempts to create cognitive systems able to recognize objects and events in a given area of interest. However, presented approaches tend to be limited to its own specific areas of interest and scenario dependence limited to arguably very small application domains ([Thonnat, 2008], [Maillot, 2004], [Auer, 2005], [Ling, 2012], [Nieto, 2014], [Bauckhage, 2004], [Lim, 2014], [Jung, 2013]).

Some reported works aim to improve the way detection and semantic annotation of events in videos takes place ([Thonnat, 2008], [Maillot, 2004], [Ling, 2012], [Long and Jung, 2015]). A consensus exists that spatial–temporal detection of objects is required for effectively identifying events and minimizing false alarms [Little, 2013]. We follow this principle as we track intra-frame detections over time analyzing also their spatial information. Development of new algorithms for event recognition seems to be most common ([Thonnat, 2008], [Little, 2013]). With our approach, we attempt to make possible to implement those algorithms within our platform.

In order to manage more meaningful detections, contextual areas can be defined to indicate parts of the scene where specific events can take place and be expected [Fritsch, 2003]. We concur with the aforesaid technique in our approach, narrowing, in this way, some of the possible events to be detected to certain areas. With additional contextual information, such as the scene perspective or prior information of the expected type of objects in the scene, it is also possible to dramatically enhance the speed and accuracy of detection algorithms [Nieto, 2014]. In other cases, authors present advances in the development of computer vision parallel with the development of event recognizers. It is always necessary to work in both areas to further progress in the cognitive vision domain. Sadly, currently presented proposals suffer from scalability issues. It is common that in order to make improvements or adapting to new (not originally considered situations) a large amount of the code has to be re-written. This happens because object recognition and event recognition are not differentiated.

The approach we propose has been designed to simplify the scalability problems mentioned before by implementing a flexible framework for video event recognition. Following Thomas Gruber's definition of what ontology is: the explicit specification of a conceptualization [Gruber, 1995]; our system is ontologically-extensible, providing the resulting knowledge to additional reasoning capabilities in the semantics data.

2.1 The Set of Experience Knowledge Structure (SOEKS) – Modeling the User Experience via a Semantic approach.

Set of experience has been developed to store formal decision events explicitly [Sanín, 2012]. It is a model based upon existing and available knowledge, which must adjust to the decision event that it is built from. Four basic components in the SOEKS surround decision-making events: variables, functions, constraints, and rules. They are stored in a combined dynamic structure that comprises set of experience. Variables usually involve representing knowledge using an attribute-value language (that is, by a vector of variables and values). This is a traditional approach from the origin of knowledge representation, and is the starting point for the set of experience. Variables that intervene in the process of decision-making are the first component of the set of experience. These variables are the root of the structure, because they are the origin of the other components. Based on the idea of Malhotra [Sanín, 2012] who maintains that "to grasp the meaning of a thing, an event, or a situation is to see it in its relations to other things", variables are related among them in the shape of functions. Functions, the second component, describe associations between a dependent variable and a set of input variables; moreover, functions can be applied for reasoning optimal states, because they result from the goals of the decision event.

Therefore, set of experience uses functions, and establishes links among the variables constructing multi-objective goals. According to Theory of Constraints (TOC), Goldratt [Goldratt, 1986] affirms that any system has at least one constraint; otherwise, its performance would be infinite. Thus, constraints are another form of relationships amongst the variables; in fact, they are also functions.

A constraint, as the third component of set of experience, is a restriction of the feasible solutions in a decision problem, and a factor that limits the performance of a system with respect to its goals. Finally, rules are suitable for associating actions with conditions under which the actions should be performed. Rules, the fourth component of set of experience, are another form of expressing relationships among variables.

They are conditional relationships that operate in the universe of variables. Rules are relationships between a condition and a consequence connected by the statements IF-THEN-ELSE. In conclusion, the set of experience consists of variables, functions, constraint and rules, which are uniquely combined to represent a formal decision event. Sets of experience can be used in platforms to support decision-making, and new decisions can be made based on such sets. SOEKS is able to collect and manage explicit knowledge of different forms of formal decision events [Sanín, 2012]. In the SOEKS, the four basic components (see Figure 1), variables, functions, constraints and rules are associated and stored in a combined dynamic structure that allows a cognitive vision system to greatly benefit from stored experience.

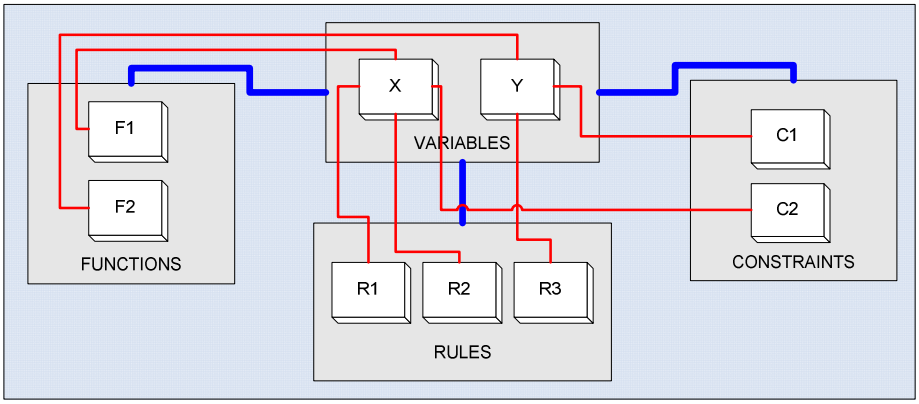

Figure 1: The Set of Experience Knowledge Structure

The combination of the four components of the SOEKS offers distinctiveness due to the elements of the structure that are connected among themselves imitating part of a long DNA strand. A SOEKS produces a value of decision, called efficiency. Besides, it is possible to group sets of experience by category, that is, by objectives. These groups could store a “strategy” for such category of decisions.

3 Approach Overview

Our goal is to support the speed of development processes in cognitive vision. We define Cognitive Vision (CV) as an attempt to achieve more robust, resilient, and adaptable computer vision systems by endowing them with a cognitive faculty: the ability to learn, adapt, weight alternative solutions, even the ability to develop new strategies for analysis and interpretation and anticipate the occurrence of objects or events. In essence, CV is a combination of computer vision and cognition ([Auer, 2005], [Vernon, 2008]).

Computers, unfortunately, are not yet capable of forming representations of the world in this way, and even simpler, representations of just formal decision events. This presents a problem of how to adequately, efficiently, and effectively represent information and knowledge inside a computer. For this reason, our consideration of using SOEKS as carrier for decisions making is logically founded in the fact that experience has to be taken into account for developing a better cognitive vision system. In regards to the implementation of SOEKS being the base of our system, out of the four components of the SOEKS, variables and constraints are easy to be understood in the context of computer vision, and functions are implemented in a programmatic perspective; then, we focus this paper on the production rules as it is in this context where an expert would generate the prerogatives for the conceptual model.

3.1 An architecture for Cognitive Vision based on run-time production rules

With so many diverse approaches to achieve computer recognition of human activities [Aggarwal, 2011], it has become necessary to effectively separate and reuse the logic that gets implemented for event recognition.

Our proposed approach is based on a service-oriented model. It handles the logic used for event recognition in videos, using object detection from computer vision data. This improves the production rate of software development as the logic for event recognition can be handled separated from the logic for object recognition. Allowing building, managing and applying reusable components easily and efficiently [Zhu, 2005]. Currently, we use boundary detection for events to be recognized. The reason is that they provide a generic segmentation of the video, which is not dependent on the action classes [Weiland, 2011]. Therefore, software will no longer be developed, integrated, and released in a centrally synchronized way. Instead, developers will be able to concentrate on improving their computer vision programs and choosing the required production rules for their implementations [Lee, 2010]. That is, if the rules already exist (see section 3.2). To validate our approach, we have developed a tool with the flexibility to help recognize events in videos while being non-dependent on the scenario. Every client sends their own set of rules (which describe the events they are interested in), the object detections obtained from videos and the parameters that govern the analysis of the detections. As a result, the workload required for adapting the programs to each situation and/or user is greatly reduced. By giving the tool additional information (e.g. height, width, frame rate of the video, etc.), it is possible to use them in the rules, achieving more accurate results. Since we are trying to semantically annotate events, we will be using the rules for describing the context of the video. Physical objects, both contextual and mobile are described with attributes.

Events are described by the evolution and interaction of an object in the scene [Thonnat, 2008]. Thus, the context of an action describes its necessary preconditions and its effect on the scene while the evolution of an object in the scene relates it with another object through interaction [Bauckhage, 2004].

3.2 An architecture for semantic annotation of video using production rules

In this section, we present an architecture centred on event detection. It consists of three layers: Data Layer, Information Layer and Knowledge/Experience Layer. Our architecture is divided in two different sections: one for the Client and the other for the Server (Figure 2 depicts the architecture from a conceptual perspective).

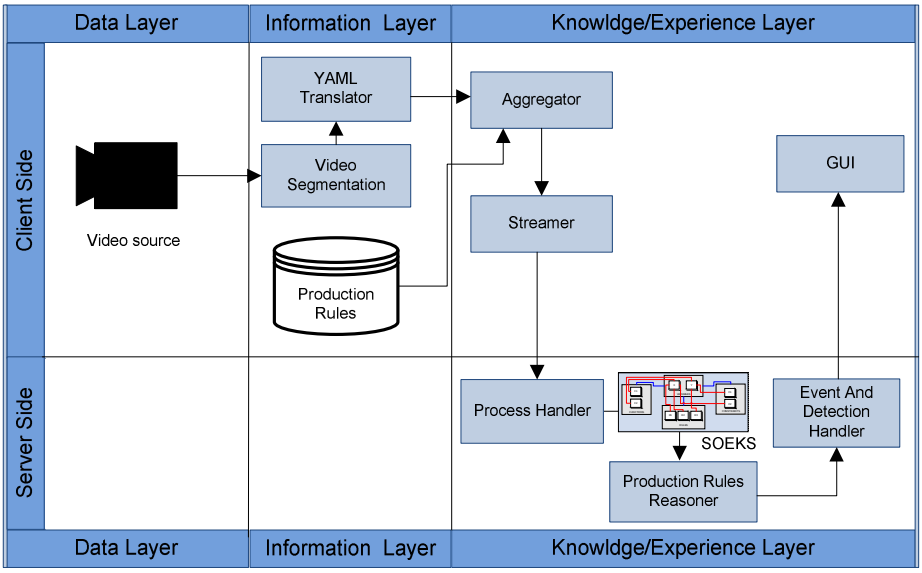

Figure 2: Cognitive Vision architecture

Since the logic is based in rule programming (and contained in the SOEKS structure), developers are able to separate conditions for event recognition as in the form of methods. With this approach, they are granted great flexibility for permanently changing conditions from a user perspective without recompiling the whole application.

Data layer: This layer consists on the source of required data for the architecture to work. The basis of all video analysis is a good source of video. Any source should work, but in our case, the tests were performed with security cameras located at fixed positions. Any camera movement was suppressed to facilitate testing, although theoretically this could be handled through rules (see section 3.2).

Information Layer: In this layer, the Video Segmentation module receives videos from security cameras. In this module, the videos are analyzed frame by frame

for object recognition. To name a few, the data gathered from detected objects are: (i) frame of detection, (ii) type of object, (iii) position, (iv) dimensions and (v) variation from the previous frame. This module also handles the basic information of the video, e.g. width, height and frame rate. That information can be used within the rules for more accurate results. The video information and detections go into the YAML Translator module where they are exported in YAML format. YAML is a recursive acronym for "YAML Ain't Markup Language". Early in its development, YAML was said to mean "Yet Another Markup Language". YAML was chosen for its (i) readability, (ii) portability between programming languages, (iii) expressiveness (iv) extensibility and (v) ease of implementation and usage [Ben-Kiki, 2009]. Due to rule simplicity, they can be stored as plain text files and/or databases. Regardless of the storage method, rules are transferred to the server at runtime. To respond to changing situations, it is possible to have different text files with rules and select those that are desired for a given requirement. In the event of having them stored in a database, it is also possible to implement a system for rule selection for increased flexibility.

Knowledge/Experience layer: In the first module of this layer (Aggregator), the production rules are joined with the detections and configuration information on the client side. The collected data is then transferred into the Streamer module, which sends it in fragments to the server. On the server side, the Process Handler opens first a dedicated connection for each client and redirects the data streamed from the client into the next module, the *Production Rules Reasoner*. This module uses each client's rules to identify those events that are of interest to the client. For the sake of accuracy, the production rules engine uses the frame number (time) and position/shape (space) of every detected object to identify events existing in the video. The spatiotemporal analysis plays a significant role in event detection and recognition [Ling, 2012] and reducing the number of false alarms [Little, 2013]. As the reasoner detects a new event, it is transferred to the Event and Detection Handler.

The last module in the server-side is used for semantically annotating the detections of objects and events. A possible scenario for the exploitation of semantic data by ontology based technologies [Toro, 2008]. After the annotation process, each detection is sent back to the client. Back on the client-side, the GUI handles the data by showing it to the user. After the server finishes processing the data and sends back the results, the user can export the results for further analysis. As for the time this work was written, there were two options for exporting; (i) Viulib (a set of libraries for computer vision applications) and (ii) ViPER (Visual Information Processing for Enhanced Retrieval) format, both being XML based. Nonetheless, other export/import formats can be added as plugins. At this point, it is up to the client to choose whether to edit or change the used production rules or change the video to analyze. This layer contains the SOEKS as container for the decisions and interacts with the knowledge layer through the reasoner, as SOEKS can be evolved using evolutionary inspired techniques; the evolution of the experience model is assured. In this paper, we focus on the production rules which are the key places where a surveillance expert will provide his knowledge and experience to the cognitive vision system.

3.3 Example of a simple production rule contained in SOEKS

The production rule engine in the SOEKS keeps our approach flexible; provided such engine maintains the underlying logic model. Rules describe the events of interest and can be written in Java. Rules act in accordance to with the detections that have already been made in the video stream using computer vision techniques and can be created and updated from outside the code in execution time with any text processor. Figure 3 depicts an example of a rule.

```
rule "Am I a Persona?"
 agenda-group "detection"
 salience 250
 when
  $aO : AbstractObject (
   type == "Unknown" ,
   $id : id ,
   averageHWRatio >= 1.85
    && <= 2.15
  );
 then
  modify ( $aO ) {
   setType ( "Person" )
  };
  update ( $aO );
end
```

Figure 3: Example of a production rule

Rules can be divided in groups for declaring "global" priorities. This is called agenda-groups being the SOEKS equivalence to decisional chromosomes. Following that action, the "local" priority within each group is taken into consideration, which is called salience. It is interesting to note that the higher the value of salience, the higher the priority of the considered rule. All the rules will be triggered as soon as their conditions are met. In the given example, the rule will be triggered once an unidentified object with an average height-width ratio between 1.85 and 2.15 (square with approximate height and width ratio of 2:1) appears. The second part of the rule is the consequence clause, which can be controlled. For our example, such object will be considered as a "Person". The level of simplicity or complexity of a rule will be determined by the user and the involved scenario. The higher the complexity of the events to detect will give more complex rules (combination of rules) which should be separated into simpler rules (single rules). It is usually possible to split a complex rule into several separated and simpler rules.

Rules as part of SOEKS are managed by the server side application of choice for rule processing (in our case Drools [De Ley, 2011]). The aforementioned fact allows us to be able to modify on the fly such rules and store them in manageable simple plain-text files containing the sets of rules. At runtime, previously selected rules are transferred to the server. Every file can keep its own set of rules to be used for diverse situations. Since rules are stored in the SOEKS in plain text, they can also be stored in

databases. At runtime, a text file is simulated with the chosen rules and sent to the server.

Within the rules, the semantics used to describe each object may be defined to adapt for the required context, which plays a fundamental role in the recognition of complex events [Fernández, 2011]. These descriptions are the output annotations of the program.

3.4 Example execution

During the developing of this work, several videos related to surveillance applications were used for testing. They were first processed with blob detection methods. This process uses a combination of full and upper-body detectors with perspective information of the scene [Nieto, 2014]. In these videos, several different types of actors could be seen, e.g. people, vehicles or just faces at a doorstep. Our application in conjunction with the service was capable of seamlessly work with these diverse videos. Figure 4 depicts the transmission of data to the server.

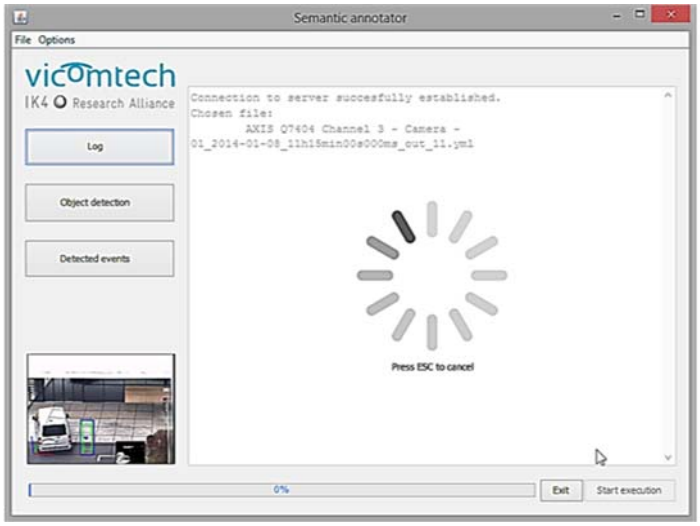

Figure 4: Screenshot of the application while sending the detections to the server

The tool has the flexibility to process and analyze any type of detection, as long as the user uses rules capable of handling the detections of the video. Among the detected events, we can find people walking or running, groups of people or a person leaving a vehicle, etc. For this case, the same rules where applied for diverse videos with different interests. The steps taken for each test are:

1. Connect to the server after the tool starts and receive confirmation.
2. Search for a YAML detection file with the tool.
3. Start execution.
4. Once the analysis is completed, a detection summary is presented to the user on the log as can be seen on Figure 5. The fields shown are:

(a) Lapsed time for the analysis (once the detections have been transferred to the server).
(b) Number of detections by type of objects.
(c) Number of events by type of event and the object associated.

5. In the "*Object detection*" section, details about every detected object can be found; e.g. to name a few: type, frame start, frame end.
6. In the "*Detected events*" section, details about every detected event can be found; e.g. to name a few: type, actors, description. Figure 5 depicts a close-up of some of the results of our tool.

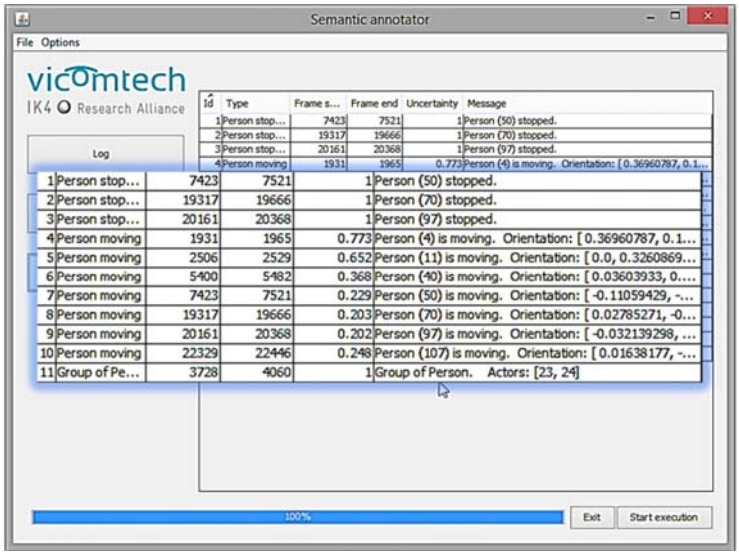

Figure 5: Details of the detected events

After the server has finished the execution, the user is be able to export the results in either Viulib [Vicomtech, 2014] or ViPER [ViPER, 2003] XML formats for further analysis or database storaging.

3.5 Evaluation of results

The proposed tool has been applied under a variety of sequences representative of different kinds of scenario relative to surveillance applications. This section summarizes the undertaken rule design activities we have carried out to show the flexibility of our system. For the tests we have used several sequences ("Indoor Access" (BISAIA project), "Building Entry" and "Parking" (FP7 SAVASA project), and "Sterile Zone" [i-LIDS, 2008], which demonstrate different capabilities of the system. A common set of base rules have been used for all the sequences, which provide basic analysis capabilities such as identifying objects by their type, grouping them by proximity or labelling their motion. Additional specific rules have been created for describing the particularities of each sequence.

1) “Indoor access”: this is a sequence of 50000 frames (about 30 minutes), showing the entrance of a cantina at lunch time. People appear from the corridor, stop at the entrance to validate their ID, and enter the room. Some people cross the corridor from side to side without entering the room, and some people get out the room in opposite direction, with a total number of 160 people.

This scenario has been included to show the application of conversion rules. In this case, we have used face detection and upper-body detection algorithms from OpenCV, which generate “Face” objects. One of the rules we use transforms a “Face” object into a “Person” object in order to enable other rules based on person dynamics. As explained along the paper, the rules can be made as complex as desired. For this experiment, we kept them simple enough: right-to-left movement is considered as “Entering” and left-to-right is “Exiting” which in turn yields a number of false alarms and false negatives due to occlusions between people or incorrect face detections. As a result, Table 1 summarizes the entering/exiting event detection.

Table 1. Recall (R), Precision (P) and F-measure (F) of the events of “Indoor access”.

Event	GT	TP	FP	FN	R	P	F
Person Entering	152	132	52	20	0.868	0.717	0.785
Person Exiting	4	1	3	3	0.250	0.250	0.250
Group of People	11	8	6	3	0.727	0.571	0.640

2) “Building entry”: a person is walking outside the main entrance of a building, talking by phone and wandering around. This person enters and exits the building a number of times (Figure 7).

Figure 6: Detection of People and estimation of their trajectory to determine “Person Enters” and “Person Exits” events.

For this scenario, we have manually annotated the event “Person Walking”, with starting and end frame metadata; and run a person tracking algorithm based on motion detection [Nieto, 2014]. We then feed the proposed tool with an “Object Moving” rule and obtain the results.

The comparisons can be made at event level, considering a true positive (TP) a detected event whose time interval approximately matches with the time interval of a ground truth. However, the usage of noisy detections typically results in several many shorter events for a ground truth long event. Therefore, we switch to a frame-wise analysis, where for each frame of the sequence we look for matching exiting events and generate TP when a match is found, a FP when detection has not a ground truth matching, and a FN.

The following Table shows the obtained values of Recall-Precision. As we can see, the numbers are pretty high in this case given that the event is rather simple.

Event	GT	TP	FP	FN	R	P	F
Person Moving	7563	7240	939	323	0.957	0.885	0.919

Table 2: Recall (R), Precision (P) and F-measure (F) of the events of "Building entry" with analysis at frame level.

3) "Parking": this sequence shows some vehicles parking in a parking area, and the drivers getting out and walking into the building along with other pedestrians. This scenario shows the capabilities of the system to cope with different type of objects and correctly identify interactions between them. The detections of such types can be provided by different sensor systems and fused together in the proposed tool through the execution of the rules. In this example, we used two different video analytics to detect vehicles and pedestrians.

The rule identifies the relative time and spatial coherence between such detections and creates the "Person Out of Car" event. From the point of view of forensic applications, such type of events can be extended or modified in order to know whether a driver has actually got off the car or remains inside, which can be a symptom of suspicious activities depending on the context (see Figure 7).

Figure 7: Detection of Cars and People can be used to create interaction rules, such as "Person Out of Car".

4) Sterile zone [i-LIDS, 2008]: this is a dataset containing images of people approaching a sterile zone with a fence they break and pass through. Some people walk very slowly, others run, and some others crawl or •roll on the floor (see Figure 9). This scenario is of particular interest for surveillance applications, where a sterile zone is defined during setup of the system, and then incorporated to the rule files. There is no need to add such information at the video analytics level and can be kept as simple as required. In our case, we used a blob analysis.

The rules incorporate an analysis of the aspect ratio in order to determine whether the event is a "Person Walking" or a "Person Crawling" or "Person Rolling".

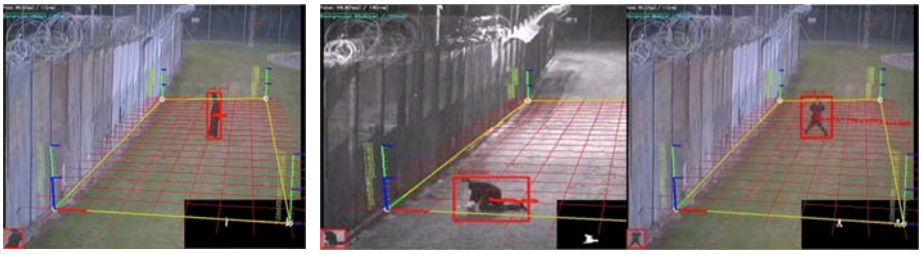

Figure 8: Blob detection can trigger launching "Person Walking", "Person Crawling" or "Person Running" events in surveillance applications.

4 Conclusions

In this paper, we have presented an automatic semantic annotation service that has the flexibility to work in diverse video analysis scenarios requiring little or no modification. Our architecture allows the reutilization of the service every time the scenario changes by using a different set of rules (whenever necessary) at runtime. For improved result accuracy, a greater detail of the context can be provided (e.g. areas of interest, expected sizes of objects, etc.). In our approach, we particularly focused in solving some of the reported scalability issues found in current Computer Vision approaches by introducing an experience based approximation based on the Set of Experience Knowledge Structure. We believe that the results presented indicate that our approach can be used as a horizontal tool that could help in automating some of the most tedious tasks that could be presented to a computer vision programmer.

5 Future Work

At the time of writing, our ideas have been implemented in a tool aiming off-line video analysis and indexing for events in security videos. In near future, we expect to be able to report detected events in real time. We have already foreseen the implementation of live streaming from cameras. By using other sensors, like microphones, video detections can be supported and event analysis will be more specific and meaningful, to be used in other areas that require greater sensory stimulation. Rules can be optimized if integrated with machine learning. Therefore, an initial training would be required for the server to learn and improve its accuracy in event recognition. This will increase the program´s adaptability without the need of constant human intervention.

References

[Aggarwal, 2011] Aggarwal, J. K., Ryoo, M. S.: Human activity analysis: A review, ACM Computing Surveys, vol. 43, no. 3, pp. 1-43, 2011.

[Auer, 2005] P. Auer, et al.: A Research Roadmap of Cognitive Vision, ECVision: European Network for Research in Cognitive Vision Systems, 2005. [Online].

[Ben-Kiki, 2009] Ben-Kiki, O., Evans, C., döt, Net, I.: YAML Ain't Markup Language (YAML™) Version 1.2, 2009. [Online]. Available: http://www.yaml.org/spec/1.2/spec.html.

[Bauckhage, 2004] Bauckhage, C., Hanheide, M., Wrede, S., Sagerer, G.: A cognitive vision system for action recognition in office environments, 2004.

[De Ley, 2011] De Ley, E., Jacobs, D.: Rules-based analysis with JBoss Drools: adding intelligence to automation, Contributions to the Proceedings of ICALEPCS 2011, Ghent, Belgium 2011.

[Fernández, 2011] Fernández, C., Baiget, P., Roca, F. X., González, J.: Determining the best suited semantic events for cognitive surveillance, Expert Systems with Applications, vol. 38, no. 4, pp. 4068-4079, 2011.

[Fritsch, 2003] Fritsch, J. N.: Vision-based recognition of gestures with context, Bielefeld University, Bielefeld (Germany), 2003.

[Goldratt, 1986] Goldratt, E. M., Cox, J.: The goal. Technical report, Grover, Aldershot, Hants, 1986.

[Gruber, 1995] Gruber, T. R.: Toward principles for the design of ontologies used for knowledge sharing, International Journal of Human-Computer Studies, vol. 43, no. 5-6, pp. 907-928, 1995.

[i-LIDS, 2008] Home Office Scientific Development Branch. Imagery library for intelligent detection system i-LIDS. http://scienceandresearch.homeoffice.gov.uk/hosdb/cctv-imaging-technology/video-based-detection-systems/i-lids/.

[Jung, 2011] Jung, J.J.: Service Chain-based Business Alliance Formation in Service-oriented Architecture, Expert Systems with Applications, vol. 38, no. 3, pp. 2206-2211, 2011.

[Jung, 2012] Jung, J.J.: Evolutionary Approach for Semantic-based Query Sampling in Large-scale Information Sources, Information Science, vol. 182, no. 1, pp. 30-39, 2012.

[Jung, 2013] Jung, J.J.: Contextual Synchronization for Efficient Social Collaborations in Enterprise Computing: a Case Study on TweetPulse, Concurrent Engineering-Research and Applications, vol. 21, no. 3, pp. 209-216, 2013.

[Lee, 2010] Lee, J., Muthig, D., Matthias, N.: A feature-oriented approach for developing reusable product line assets of service-based systems, Journal of Systems and Software, vol. 83, no. 7, pp. 1123-1136, 2010.

[Lim, 2014] Lim, M. K., Tang, S., Chan, C. S.: iSurveillance: Intelligent framework for multiple events detection in surveillance videos, Expert Systems with Applications, vol. 41, no. 10, pp. 4704-4715, 2014.

[Ling, 2012] Ling, S., Ling, J., Yan, L., Jiagou, Z.: Human action segmentation and recognition via motion and shape analysis, Pattern Recognition Letters, vol. 33, no. 4, pp. 438-445, 2012.

[Little, 2013] Little, S., Jargalsaikhan, I., Clawson, K., Li, H., Nieto, M., Direkoglu, C., et al: An Information Retrieval Approach to Identifying Infrequent Events in Surveillance Video, Dallas, 2013.

[Long and Jung, 2015] Long, N.H., Jung, J.J.: Privacy-aware Matching Online Social Identities for Multiple Social Networking Services, Cybernetics and Systems, vol. 46, no. 1-2, pp. 69-83, 2015.

[Maillot, 2004] Maillot, N., Thonnat, M., Boucher, A.: Towards ontology-based cognitive vision, Machine Vision and Applications, vol. 16, nº 1, pp. 33-40, 2004.

[Nieto, 2014] Nieto, M., Ortega, J. D., Cortes, A., Gaines, S.: Perspective Multiscale Detection and Tracking of People, Lecture Notes in Computer Science, vol. 8326, pp. 92-103, 2014.

[Sanín, 2012] Sanín, C., Toro, C., Haoxi, Z., Sanchez, E., Szczerbicki, Carrasco, E., et al.:Decisional DNA: A multi-technology shareable knowledge structure for decisional experience Journal of Neurocomputing, 88 , pp. 42–53, 2012.

[Thonnat, 2008] Thonnat, M.: Semantic Activity Recognition, in Proc. 2008 Conference on European Conference on Artificial Intelligence, pp. 3-7, 2008.

[Toro, 2008] Toro, C., Sanín, C., Szczerbicki E., Posada J.: Reflexive Ontologies: Enhancing Ontologies with self-contained queries, Cybernetics and Systems: An International Journal, vol. 39, no. 2, pp. 171-189, 2008.

[Vernon, 2008] Vernon, D.: Cognitive vision: The case for embodied perception, Image and Vision Computing, vol26, no. 1, pp. 127-140, 2008.

[Vicomtech, 2014] Vicomtech-IK4: Viulib, 2014. [Online]. Available: http://www.viulib.org/.

[ViPER, 2003] Language and Media Processing Laboratory, University of Maryland, "ViPER: The Video Performance Evaluation Resource," 2003. [Online]. Available: http://viper-toolkit.sourceforge.net/.

[Weiland, 2011] Weinland, D., Ronfard, R., Boyer, E.: A survey of vision-based methods for action representation, segmentation and recognition, Computer Vision and Image Understanding, vol. 115, no. 2, pp. 224-241, 2011.

[Zheng, 2014] Zheng, X., Yunhuai, L., Chuanping, H., Lan, C.: Semantic based representing and organizing surveillance big data using video structural description technology, Journal of Systems and Software, in press, 2014.

[Zhu, 2005] Zhu, H.: Building reusable components with service-oriented architectures, 2005.