OBJECT-ORIENTED QUERY LANGUAGE FOR EVENTS DETECTION FROM IMAGES SEQUENCES

GANEA Ion Eugen

Assistant professor, Faculty of Automation, Computers and Electronics/Department Computers and Information Technology, University of Craiova, Romania, ganea_eugen@software.ucv.ro

Abstract: *In this paper is presented a method to represent the events extracted from images sequences and the query language used for events detection. Using an object oriented model the spatial and temporal relationships between salient objects and also between events are stored and queried. This works aims to unify the storing and querying phases for video events processing. The object oriented language syntax used for events processing allow the instantiation of the indexes classes in order to improve the accuracy of the query results. The experiments were performed on images sequences provided from sport domain and it shows the reliability and the robustness of the proposed language. To extend the language will be added a specific syntax for constructing the templates for abnormal events and for detection of the incidents as the final goal of the research.*

Key words: event representation, semantic video interpretation, query language, event detection

1. Introduction

The growing of video data and the expansion of video surveillance systems requires new approaches for the events detection domain. This paper describes the research done to build an object-oriented language that allows storage and querying of video events. The main goal of the presented work is to construct a unified language for events representation and also for querying them to extract new types of events. The main phases needed to achieve this are (a) an object-oriented model for video processing; (b) usage of the syntactic and semantic features of the salient objects detected in the images sequences for the spatial and temporal relationships representation as rules for an inference engine; (c) usage of the declarative component added to the object-oriented model for efficient events detection based on the interactions between salient objects and between sub-events. As definition: an event is an observed physical reality parameterized by space and time [1]. The observations describing the event are defined by the syntactic and semantic properties of the salient objects. The syntactic features are translated in the attributes of the classes from object-oriented model and the semantic behaviour is implemented as messages changed between instances of these classes. In this way using the object-oriented paradigm are built the patterns for the event as class hierarchy. The methods used for events detection based on a descriptive language are mainly based on two techniques: the usage of the natural language to describe the scenes using the semantic features and the formal languages constructed such that their syntax allows specifying the complex relationships from videos data.

In [2] is described VERL standard for Video Event Representation Language. This language was used to generate an event ontology using the standard Web Ontology Language (OWL) [3]. VERL is used to describe event ontology and it requires the full OWL representation. A related work is presented in [4] as EDF, the Event Description Framework. EDF language is oriented on a relational representation of events so that queries can be processed using SQL and can take advantage of the availability of spatial indexes and reasoning capabilities that have been developed by the relational database community. The Petri nets formalism is used for events specifying by a transition [5], where the type of node that represent the transition depends on the event type (primitive or composite). One of the positive aspect of using Petri nets is that it can be used for both deterministic and fuzzy inference of event occurrences. In [6] events from images sequence are described using the context-free grammars; it used the classification entropy as a heuristic function and the grammars are iteratively learnt using a search

method. A search-based iterative algorithm for constructing the grammar structure and members for each category of motion is developed in a semi-supervised learning strategy. For syntactic pattern recognition in [7] the data is specified with a set of terminal symbols from an alphabet. For event detection, the terminals correspond to the primitive events extracted from the video. The abnormal events are detected when the input does not respect the grammar syntax or for attributes grammar the some constraints do not satisfy. In [8] is presented a method for learning primitive action and role models based on easily available natural language video descriptions. The proposed language topic model is based semantic relatedness measure to identify positive and negative training examples. An expert system bases a fast inference engine is presented in [9] to describe event detection. The module uses an approach which distinguishes it from other approaches in the field of video surveillance. It provides a simple syntax for rules, so that even domain experts are capable of describing events that should be detected. In [10] is proposed a spatial representation for specifying the spatial semantics of video data. Based on such a representation, a set of spatial relationships for salient objects is defined to support qualitative and quantitative spatial properties. The model captures both topological and directional spatial relationships. They describe a new method to incorporate this model into a video model, and to integrate the video model into an object database management system; the integrated model is further enhanced by a spatial inference engine. The resulted query language - MOQL (Multimedia Object Query Language) is based on Object Query Language (OQL) [11]. From this point of view this paper is the nearest research related to the current paper. As difference the proposed language - Object Oriented Event Recognition Language (OOERL) isn't so general to the multimedia information; it is specific only to the video events processing and on the other side OOERL has integrated an inference engine based on Rete algorithm [12] (a very efficient method for solving the difficult many-to-many matching problem) that allows the both spatial and temporal inference.

The structure of the paper is as follows. Section II defines the syntax of the object oriented language (OOERL) with declarative component used for the events description. Section III describes the query structure; in chapter IV are presented the results of the method for visual events detection within videos from sport domain.

2. Object-Oriented Language for Events Representation

The object oriented language developed for events representation is constructed over a rules base system as it is Jess system [13]. Jess is a rule engine and scripting environment that uses knowledge specified with the declarative rules. Jess uses an enhanced version of the Rete algorithm to process rules. Jess has many unique features including backwards chaining and working memory queries, and it can directly works Java objects. Jess allows creating Java objects, calling Java methods, and implementing Java interfaces. Based on all these features it was developed an object-oriented language that uses as support Object Data Language (ODL) standard [9].

The subset of ODL language used for events description (OOERL) is based on the following grammar:

specification	→ interface class module
interface	→ interfaceDcl forwardDcl
interfaceDcl	→ interface Id { interfaceBody } interface Id : inheritanceSpec { interfaceBody }
forwardDcl	→ interface Id
interfaceBody	→ export export interfaceBody
export	→ typeDcl constDcl attrDcl relDcl
class	→ classHeader { interfaceBody }
classHeader	→ class Id
module	→ module Id { specification }
inheritanceSpec	→ scopedName scopedName , inheritanceSpec
scopedName	→ Id :: Id scopedName :: Id
constDcl	→ const constType Id = constExp
constType	→ numberType charType booleanType stringType scopedName
typeDcl	→ typedef typeDeclarator structType unionType enumType
typeDeclarator	→ typeSpec declarators
typeSpec	→ simpleTypeSpec constrTypeSpec
simpleTypeSpec	→ baseTypeSpec templateTypeSpec scopedName
baseTypeSpec	→ numberType charType booleanType
templateTypeSpec	→ arrayType stringType collType
collType	→ set simpleTypeSpec list simpleTypeSpec

constrTypeSpec	→ structType enumType
declarators	→ declarator declarator , declarators
declarator	→ simpleDeclarator complexDeclarator
simpleDeclarator	→ Id
complexDeclarator	→ arrayDeclarator
structType	→ struct Id { MemberList }
memberList	→ member member memberList
member	→ typeSpec declarators
elementSpec	→ typeSpec declarator
arrayType	→ array simpleTypeSpec sequence SimpleTypeSpec
attrDcl	→ readonly attribute domainType attributeList attribute domainType attributeList
domainType	→ simpleTypeSpec structType set Literal list literal
relDcl	→ relationship targetOfPath ID optInverseTraversalPath
targetOfPath	→ Id relCollectionType Id
optInvTraversalPath	→ inverse Id :: Id
attributeList	→ scopedName scopedName , attributeList
relCollectionType	→ set list

To extend this syntax for video events processing and representation in an object oriented database I added the following productions rules:

classHeader	→ class Id is-a Id { role } { pattern-match }
role	→ concrete abstract
pattern-match	→ reactive non-reactive
attrDcl	→ readonly attribute domainType attributeList { facets } attribute domainType attributeList { facets }
facets	→ { access-facet } { propagation-facet } { pattern-match-facet } { visibility-facet } { index-propagation-facet }
access-facet	→ read-write read-only initialize-only
propagation-facet	→ inherit no-inherit
pattern-match-facet	→ reactive non-reactive
visibility-facet	→ private public
index-propagation-facet	→ inherit no-inherit

Each attribute of a salient object that contributes to an event achievement has a facet (propagation-facet) that allows inheriting or not the attribute in the current event description. The requirement for this facet is specific to the situation wherein the same salient object participates with some attributes in the realizing process for an event and with other part of attributes in other event.

Using the syntactic attributes properties of each salient object are annotated the links between objects with labels that specified the spatial relationships between two neighbor objects as in Figure 1. The spatial relationships are grouped in three categories: distance relations, direction relations and topologic relations.

Table 1: Spatial relationship types

Relation T	Opposite relation	Symbol	Semantics
Inside	Outside	ins (o1, o2)	salient object o1 is in the interior of the salient object o2
Outside	Inside	ots (o1, o2)	salient object o1 is outside of the salient object o2
Above	Below	abv (o1, o2)	salient object o1 is below of the salient object o2
Below	Above	blw (o1, o2)	salient object o1 is above of the salient object o2
Left	Right	lft (o1, o2)	salient object o1 is to right of of the salient object o2
Right	Left	rgt (o1, o2)	salient object o1 is to left of the salient object o2

For each topologic relation is used a rule based on the geometrical features that allow establishing the relation type; the rules specification is made with Jess language.

For inside/outside relationships were defined the following rules that are based on the values for the geometrical feature of the common frontier of two objects:

```
(defrule inside_o1_o2 (defrule inside_o2_o1
  (= perim (o1) common_perim) (= perim (o2) common_perim)
=> =>
```

<pre> ins (o1, o2)) (defrule outside_o1_o2 (<> perim (o1) common_perim) => ots (o1, o2)) For the determination of relative position relationships are used the following rules: (defrule belowAbove_o1_o2 (< yc1 yc2) (> (w1+w2) xc1 - xc2) => blw (o1, o2) abv (o2, o1)) (defrule leftRight_o1_o2 (< xc1 xc2) (> (h1+h2) yc1 - yc2) => lft (o1, o2) rgt (o2, o1)) </pre>	<pre> ins (o2, o1)) (defrule outside_o2_o1 (<> perim (o2) common_perim) => ots (o2, o1)) (defrule BelowAbove_o2_o1 (< yc2 yc1) (> (w1+w2) xc1 -xc2) => blw (o2, o1) abv (o1, o2)) (defrule leftRight_o2_o1 (< xc2 xc1) (> (h1+h2) yc1 - yc2) => lft (o2, o1) rgt (o1, o2)) </pre>
--	--

For temporal relationships are used point-interval and interval-interval relationships over frame sequences [14]. It is define the interval set as a set of consecutive frames. The typical relations for this kind of relationships are: before, during, after and respectively before, equal, meet, overlap, during, start, finish for interval-interval relationships. All these relations together with the all spatial relationships extended on the interval set are expressed as rules that will be considered in the inference process. The determined spatial relationships together with the temporal relationships are used to construct the patterns for the events and will be used also to formulate the queries.

The algorithm to determine the object oriented model for the salient objects detected and for the events also is based on the following steps:

- I. the establishment of the domain for the images sequences;
- II. the extraction of the salient objects (for this phase it is supposed as done and it is not part of the current research);
- III. the behavior identification for each salient object based on temporal relationships;
- IV. the addition of the attributes for salient objects;
- V. the definition for the attributes constraints ;
- VI. the identification of the structure on events and their classification.

The objects that are serialized in the object oriented database refer to two categories: one of them refers to the salient objects with their consistent attributes and the second refers to the detected patterns events. The information corresponding of both of them will be used in the query process.

3. Query Language Syntax

The SQL language is the common language used for the databases querying and it suppose the restricting a data set according to a certain set of criteria to get a result set. Object-oriented databases using an object oriented query language - Object Query Language (OQL) - which was standardized in [10] and to formulate queries using ODL schema language. The stored data contains data and executable code (through methods) so OQL queries can refer to such methods to get a complete and accurate results. The used query language is based on OQL and the schema for a database that used the language is defined as a class's hierarchy: $\{class_i \mid i = 1 \dots n\}$. A class is defined based on the triplet $\langle state, behavior, inheritance \rangle$, where:

- class.state ::= {attributes, relations};

- class.state.attributes ::= $\{ \langle attribute_i, type_i \rangle \mid i = 1..a, \text{ where } a \text{ is the total number of the attributes } \}$;

- class.state.relations ::= $\{ \langle relations, type_object_i \rangle \mid i = 1..r, r \text{ is the relations number } \}$.

The component class.behavior refers to the set of methods $\{method_i \mid i = 1..m, m \text{ is the total number of methods } \}$:

- class.behavior.method ::= {<parameters_list, type>} with class.behavior.method.parameters_list ::= {<parameter_i, type_i>, i = 1..p, p is the number of the parameters for the method}.

The component class.inheritance::={ class_i, i = 1..c, c is the number of the super classes for the current class}.

For specifying the queries the OOQL standard was extended for the OOERL described in the previous chapter. It defines a set of classes for object model data: (I) the scheme which defines attributes that contain a class of objects; the items can be atomic or multi-value; (II) the instance which defines a set of objects in accordance with the scheme. An object is a pair $O = \langle id, v \rangle$, where id is the object identifier, and v represent the value of the object (his attributes); (III) the functional dependencies values are used as input for the scheme to check if the values of a set of attributes determines the values of another set attributes; (IV) the object scheme that it is a triplet $\langle N, F, S \rangle$ with N - the value of scheme, F - the set of the functional dependencies for N and S - a set of subsets of N .

For the OOERL language has been implemented only the operators those refer to the query. Using unified structures involves several advantages: the usage of a unique formalism for the representation of classes and inheritance structural avoiding data redundancy (the values for sub-objects are inheriting from the super-objects); the definition of complex objects and the functional dependences; the support for the objects identification mechanism OIDs and the mechanisms that provide the multiple inheritance. Given all this aspects was done an interpreter that enables the translation of semantic-based query based on symbolic language into queries that respect OQL syntax.

These queries are translated by a second language interpreter as shown in Figure 1. In choosing this conversion on two levels we considered the following: (I) the database HyperGraphDB [13] that is the core of database management system used, lacks a proper query system; (II) specification of corresponding salient objects segmented and annotated information uses a unitary model; (III) as a future works OOERL will be tested as a query language for database HyperGraphDB.

Figure 1: The chain of queries transformation

4. Experiments and Results

The experimental results demonstrate that the method produces an optimal retrieval of the visual events from different images sequences. I used for experiments the video files of TRICTRAC dataset [15]. The video of the dataset refers to the synthetic video sequence of soccer. The queries type used is from query by example category. I am using the standard format specific to the object oriented native queries that allows to specific the queries as instances of the classes from developed object-oriented model. The template for this kind of queries is:

```

CEvent videoEventQuery = new CEvent(<<syntactic_attr_values>>);
HyperGraph goodb = new HyperGraph(<<database_location>>);
List<CFrame> frames = hg.getAll(hg.eq("syntactic_attr", <<syntactic_attr_values>>));
for (CFrame frm : frames)
    frm.ViewToUser();

```

, where goodb represents an instance of the main class HyperGraph that allows the usage of the HyperGraphDB database; hg is an instance of HGQuery class that offers the requested methods for query of the database. The classes CEvent and CFrame are part of the class hierarchy that implementing the object-oriented model.

Figure 2: Experiments Result

5. Conclusions

The current study presents an object oriented approach for representing simple and complex events and for events query language. It is proposed a unified language that uses object oriented paradigm combined with inference engine to determine and to define the spatial and temporal relationships between salient objects recognized and between detected events. The relationships and a high-level syntax for specifying events templates and query parameters are expressed as rules applied on the objects. As future works the grammar of the language will be extended to allow the specification of the syntax for constructing abnormal events templates and for the incidents detection.

6. Acknowledgments

This work was supported by the strategic grant POSDRU/159/1.5/S/133255, Project ID 133255 (2014), co-financed by the European Social Fund within the Sectorial Operational Program Human Resources Development 2007-2013.

7. References

1. Singh, R., Li, Z., Kim, P., Pack, D., Jain, R., Event-Based Modeling and Processing of Digital Media, Proceedings of the First International Workshop on Computer Vision meets Databases, CVDB 2004, pp. 19-26, (2004).
2. François, A.R.J., Nevatia, R., Hobbs, J.R., Bolles, R.C., VERL: An Ontology Framework for Representing and Annotating Video Events, IEEE Multimedia, Vol. 12, No. 4, pp. 76-86, (2005).
3. Web Ontology Language Standard, <http://www.w3.org/TR/owl2-overview/>, access in April, (2015).
4. Natarajan, P., Nevatia, R., EDF: A framework for Semantic Annotation of Video, Proceedings of the Tenth IEEE International Conference on Computer Vision, pp. 1876-1876, (2005).
5. Ghanem, N., DeMenthon, D., Doermann, D., Davis, L., Representation and recognition of events in surveillance video using Petri nets, Proceedings of workshop on event mining, vol. 7, no. 7, pp. 112, (2004).
6. Veeraraghavan, H., Papanikolopoulos, N., Schrater, P., Learning dynamic event descriptions in image sequences, Proceedings of IEEE CVPR'07, pp. 1-6, (2007).
7. Joo, S-W, Chellappa, R., Attribute grammar-based event recognition and anomaly detection, Proceedings of CVPRW'06, pp. 107-115, (2006).
8. Ramanathan, V., Liang, P., Fei-Fei, L., Video Event Understanding using Natural Language Descriptions, Proceedings of Computer Vision (ICCV), pp. 905 – 912, (2013).
9. Krausz, B., Herpers, R., Event Detection for Video Surveillance Using an Expert System, Proceeding of the 1st ACM International Workshop on Analysis and Retrieval of Events/Actions and Workflows in Video Streams (AREA 08), pp. 49-56, (2008).
10. Li, J. Z., Ozsu, M. T., Szafron, D., Modeling of Video Spatial Relationships in an Object Database Management System, International Workshop on Multi-media Database Management Systems, pp. 124-132, (1996).

11. Cattell, R. G. G., Barry, D., Bartels, D., Berler, M., Eastman, J., Gamerman, S., Jordan, D., Springer, A., Strickland, H., Wade, D., The Object Data Standard: ODMG 3.0., Morgan Kaufmann Publishers, San Francisco, CA, ISBN 1-55860-647-5, (2000).
12. Forgy, C., Rete: A Fast Algorithm for the Many Pattern/Many Object Pattern Match Problem, In Artificial Intelligence, No. 19, pp. 17–37, (1982).
13. Friedman-Hill, E., Jess in Action: Java Rule-Based Systems, Manning Publications, (2003).
14. Iordanov, B., HyperGraphDB: A Generalized Graph Database, Web-Age Information Management Lecture Notes in Computer Science Volume 6185, pp. 25-36, (2010).
15. Desurmont, X., Hayet, J. B., Delaigle, J. F., Piater, J., Macq, B., TRICTRAC Video Dataset: Public HDTV Synthetic Soccer Video Sequences With Ground Truth, Workshop on Computer Vision Based, Analysis in Sport Environments (CVBASE), pp. 92–100, (2006).