ICARUS: Intelligent Content-Based Retrieval of 3D Scene

Raffaella Colaci and Marco Schaerf

Dipartimento di Informatica e Sistemistica, Università di Roma "La Sapienza"
Via Salaria 113, I-00198
Roma, Italy
{colaci, schaerf}@dis.uniroma1.it

Abstract

We present a tool for the analysis, classification and content-based retrieval of 3D scene. The system ICARUS analyzes files in VRML format, searching for the presence of complex 3D-objects and relative geometrical relationships between them. Descriptions of the virtual scenes are classified in a Terminological System, and reasoning mechanisms are used for querying.

Keywords: virtual reality, description logics, information retrieval, geometric reasoning, knowledge representation, multimedia.

Introduction

Multimedia information systems, such as those supporting image and video databases, are more and more widespread and widely used. The particular nature of data types involved in multimedia databases might require special facilities for optimal storage, access and retrieval. An intelligent data management should provide the capability of querying multimedia object content, rather than being simply a file repository that does not understand the stored data. Multimedia data must be interpreted before it can be queried and this process demands data-specific analysis algorithms to generate content descriptions. The richness of the data model used to represent content information plays a key role in its usability. Many papers [1,6,7,8] focus on the importance of having appropriate description and querying mechanisms to describe and retrieve multimedia information. While traditional approaches use keyword indexing, we aim to show that rich object description languages can provide several advantages.

In this paper we present a prototype system, called ICARUS (Intelligent Classification And Retrieval Of Unlabelled Scene), for intelligent management of virtual scenes collections. In particular, we focus our attention on three-dimensional scenes modeled using the Virtual Reality Modeling Language (VRML). Interesting scenes represent situation on a specific domain, and knowledge of the modeled domain is used to classify and retrieve properties of the scenes.

The system purpose is to analyze VRML files, extract information on their content, and use a concept description language (CLASSIC) to describe the characteristics of the virtual scenes. Such descriptions are then classified in a conceptual structure, representing the knowledge of the domain, and reasoning mechanisms are used for information retrieving and querying.

The analysis aims to discover the presence in a virtual scene of complex 3D-objects, calculate their absolute positions in the 3D-space, and verify geometrical relationships between objects. The analyzer produces simple descriptions, while the inference engine deducts all other interesting properties about the scenes. The user does not need to specify any information. This is the reason why we call our scenes "unlabelled".

At the current status of development, ICARUS only focuses on a simple domain: a set of characters, a number of relations among them, and a set of furniture elements and accessories. Each object corresponds with a particular VRML model (prototype) that represents it in virtual scenes. The system provides the users with a rich collection of prototypes in order to model a scene. ICARUS is able to identify all the instances of the prototypes present in a file, and uses the geometrical and semantic knowledge of the models during the analysis process. In this way the system can deal with complex structured-objects, rather than only with geometric primitives.

The access to the system facilities is allowed via the World Wide Web. Users can upload their VRML files, verify the analysis results and decide whether classifying their scenes. The system provides a suitable graphical user interface for querying and data presentation.

ICARUS has been developed using the object-oriented programming languages Java. This language is well suited for network programming, in particular as it provides applets for client-side and servlets, which can replace CGI scripts, for server-side. LISP language is used for the interface to the CLASSIC KBMS, whose implementation is in LISP.

Description Logics as Data Model

At the heart of ICARUS lays its knowledge base (KB). It is defined using CLASSIC [2], a concept language in the family of Description Logics (DLs), or Terminological Logics. DLs are a fragment of first-order logic particularly well suited for specifying data classes (called *concepts*) and

relationships among classes. Moreover DLs are equipped with both a formal semantics and efficient inference mechanisms.
A distinguishing feature of DLs is that classes can be defined *intensionally*, in terms of descriptions that specify the properties that objects must satisfy to belong to the concept. In DLs-systems the KB is made up of two components: the Terminological Box (TBox) is the general schema concerning the classes, their properties and mutual relationship, and the AssertionalBox (ABox) is a instantiation of this schema, containing assertions relating individuals to classes, or individuals to each other. Formally, concepts are interpreted as sets of individuals, and *roles* are binary relations used to specify their properties. E.g., a role "`father-of`" can relate a father to his child. Functional relations are called *attributes*. A role-hierarchy is used to state that a role is a *subrole* of another one, meaning that the binary relations interpreting them are contained one in the other. E.g., one may state that "`father-of`" is a subrole of "`parent-of`".
A concept language allows the construction of composite *descriptions*, built up recursively as term from subterms, using different constructors, as boolean operators on concepts - conjunction (`AND`), union (`OR`) and negation (`NOT`). E.g., the concept `MAN` can be defined as `(AND PERSON MALE)`. Also restrictions on roles are possible: universal role quantification (`ALL`), unqualified existential role quantification, and restrictions on the number of role fillers (`AT-LEAST`, `AT-MOST`). E.g. the concept `FATHER` can be defined as `(AND MAN AND (AT-LEAST (1, father-of) ALL (father-of, PERSON)))`.
A significant feature of descriptions is that they can be reasoned with, as they form a logical theory. The fundamental logical relationship between descriptions is *subsumption* (containment): C is subsumed by B iff every possible individual instance of C would also be an instance of B. Through subsumption, concepts are put into a taxonomy (*classification*). Individuals are classified determining all concepts they satisfy.
DL-systems provide numerous other deductive services, as *consistency checking* on individuals, *incoherent concept detection*, information *propagation* and *rule application*.
DLs are particularly useful as query languages for structured data [3]. Both intensional and extensional knowledge can be retrieved. One can detect whether a query is incoherent, i.e. that cannot possibly return any individuals because of the semantics of the KB, or a query is coherent but it returns an empty set as answer because there is no known individual that satisfies all the given constraints.
CLASSIC language uses only a subset of the above possible language constructors - it does not use `OR` and `NOT` constructors - even if it offers others useful ones, such as `SAME-AS` and `TEST`. As a matter of fact CLASSIC project philosophy prefers more restricted expressiveness, but more efficient reasoning. When efficient systems, based on much more powerful DLs, will be available, ICARUS would adopt them in place of CLASSIC.

ICARUS

Architecture of ICARUS

ICARUS has a client-server architecture, as shown in figure 1. It allows using system capabilities on a remote host connected via the Internet.
We can recognize the following different components:

- An HTML Browser Java-compatible, with a plug-in capable to display VRML files;
- A Java Applet, that implements the graphical interface for querying and that allows the connection to servers;
- A Web Server, providing system access;

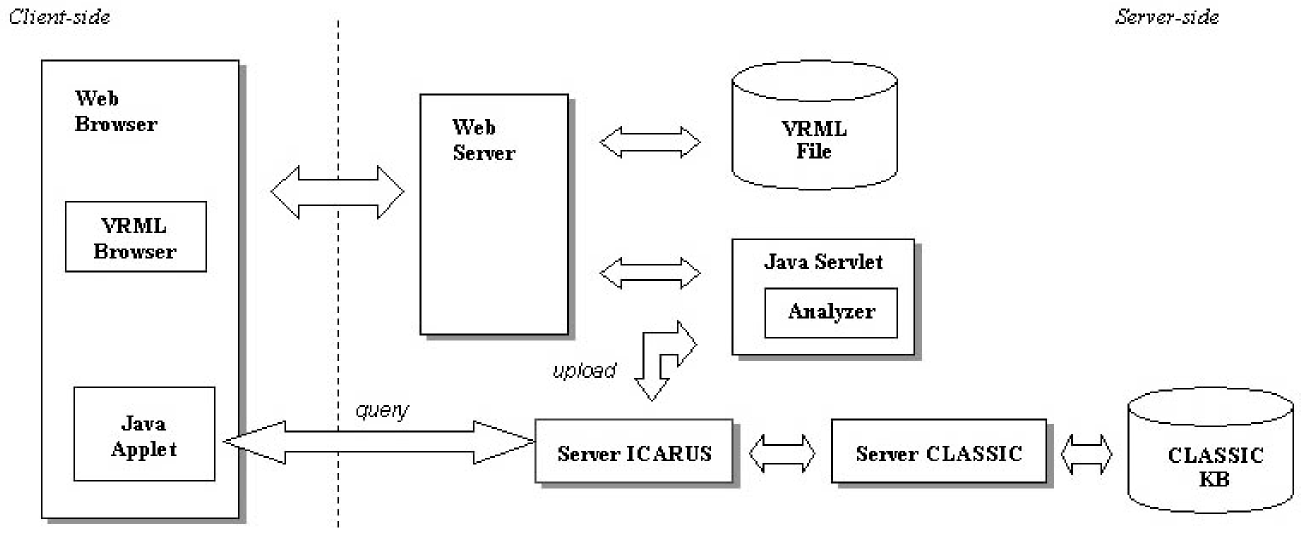

Figure 1. Architecture of ICARUS

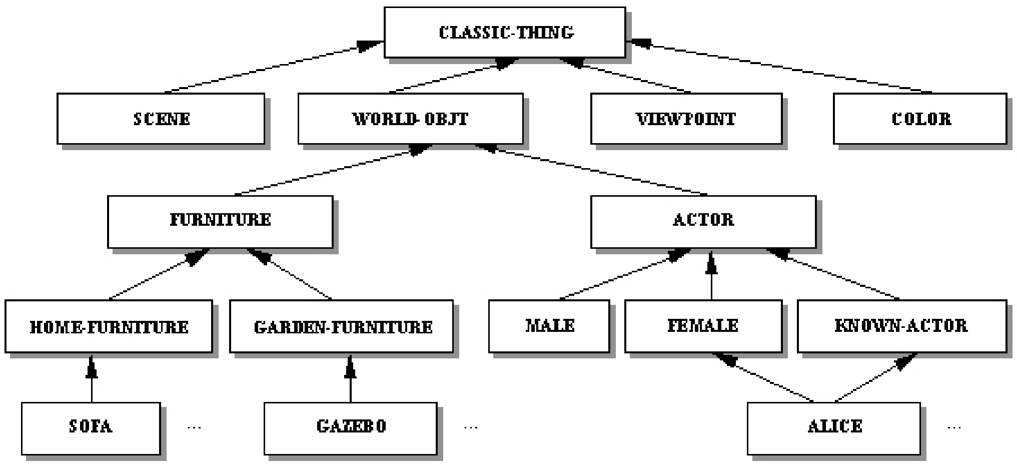

Figure 2. General conceptual structure

- A Java Servlet that the Web Server loads to manage user requests. In particular, the servlet is able to complete the upload session running the Analyzer, which parses files and produces scene descriptions for the KB.
- A Java Server, called Server ICARUS, is connected with the servlet for upload process and with the clients for querying the KB. It translates queries in CLASSIC format and synchronizes accesses to the KB.
- A LISP Server, called Server CLASSIC, is the interface between the CLASSIC KB and the other components;
- The CLASSIC KBMS that defines the structured concepts and stores the descriptions of the analyzed scenes.

ICARUS does not need special hardware features, even if users hosts should have a graphic engine support for an optimal visualization of 3D scenes. Internet connection should be as fast as possible to gain efficiency.

Software compatibility is needed. In particular the Java servlet and the LISP server should be run on the same host, in order to share files information. This should not be a problem because of the Java language is architecture-neutral and Java software platforms are ported onto various hardware-based platforms. Java 2 Platform, Standard Edition, version 1.2.2, is used. Moreover Java3D and Java Servlet packages are necessary.

For client-side, notice that HTML browser must be able to accept cookies, its JVM must be update at least at version 1.7, and it must have installed a VRML plug-in.

The Knowledge Base

Currently, the application domain of ICARUS is quite simple. The system focuses on scenes, which involve furnished environments (houses or gardens) and a set of characters, male and female (Alice, Bob, Charlie, etc.), whose physical features (skin, hair, beard colors) and clothes (shirt, skirt, trousers, shoes colors) are known. Also relations among actors are defined. E.g. Alice is a relative of Bob, friend of Diana, colleague of Charlie and loves Frank, and so on. The ICARUS KB models this knowledge of the domain. In particular every piece of furniture and character matches a specific concept. Individuals identified during file analysis process, and whose properties and geometrical information are maintained, populate the KB. The domain is described through a concept hierarchy, whose simplified view is shown in figure 2.

Relations between individuals are represented as roles. E.g. the roles `furnishing` and `characters` relate a scene, respectively, with its furniture and actors. Geometrical information, such as dimension, position and orientation are stored for every object in a scene. Roles representing geometrical relation between `WORLD-OBJECT` (`in-front-of`, `behind`, `over`, `beside`, and so forth) are also defined.

Roles have a hierarchical structure too. As we suppose love implies friendship, role “lover-of” is defined as subrole of “friend-of”. A particular use of role hierarchy allows relating specific information within a role. For example, one can join a scene to a male actor through the role `character-male`, instead of using its parent role `character`. With this structure of roles, it is possible to define a special form of qualified existential role quantification. So the concept `SCENE-AT-LEAST-1-MALE` can be expressed as `(AND SCENE (AT-LEAST 1 character-male))`. Using simply not-structured roles, we can specify only universal role quantification, such as `SCENE-ALL-MALE = (AND SCENE (AT-LEAST 1 characters)(ALL characters MALE))`.

Inverse roles are possible: for symmetric relation one can define a role and its inverse as the same (`near-near`), otherwise a different inverse name is allowed (`inside-outside`).

Some relations among concepts, such as friendship between `ALICE` and `ELIZABETH`, can be expressed using roles. In this case, since a role cannot relate two concepts, a role joins one of the concepts to an individual representing the other concept (meta-individual). Using the *rules* mechanism, relations among concepts and concept meta-individuals are inferred to individuals.

Rules are also used to deduce that a generic character, when it possesses specific physical features, is a KNOWN-ACTOR, thus retrieving the other relationships.

In the KB some pairs of concepts are defined as disjoint primitive concepts (FURNITURE and ACTOR, or MALE and FEMALE), in order to show that no individual can satisfy both of them.

As the system aims to classify scenes, the concept SCENE has a particular meaning. This concept is described as a primitive concept, while a number of *defined-concept* is based on it. A defined-concept is a concept expressing both necessary and sufficient conditions for an individual to be an instance of it. This type of concept is essentially as views in databases. A considerable advantage gained with defined-concepts is that the system itself can be charged with organizing these views into a hierarchy. Notice that the relations expressed are more complex than just the class-subclass relation. In ICARUS the taxonomy of defined-concepts provides a mechanism to describe scene properties to final users. For this reason defined-concepts are used to describe special properties of the scenes or particular situation. For example, we have defined concepts representing:

– The presence/absence of known actors (SCENE-PRESENT-ALICE, SCENE-NOT-PRESENT-BOB), or pieces of furniture (SCENE-WITH-TABLE);
– The number of characters present (SCENE-NO-CHARACTERS, SCENE-AT-LEAST-3-CHARACTERS, etc.);
– The presence of particular kind of actors (SCENE-NO-FEMALE, SCENE-ALL-FEMALE, SCENE-AT-LEAST-1-FEMALE);
– The kind of environment, based on the presence of particular furniture. For example we can recognize a ROOM, if walls, floor and door are present, or a GARDEN, if there are grass and garden furniture. Also particular kind of rooms is defined: SITTING-ROOM, KITCHEN, LIBRARY, BEDROOM, OFFICE.
– Special relations among actors (SCENE-ALL-FRIENDS, SCENE-AT-LEAST-2-COLLEAGUES) or among generic objects (SCENE-ALL-OBJECTS-NEAR);
– Particular situation (job-meeting, Christmas-party, romantic scene).

A number of described concepts cannot be expressed in CLASSIC without the use of the TEST constructor, which allows LISP procedures to be used in specifying concepts. Passing a test restriction represents a sufficient condition to satisfy a concept. Tests are used to retrieve information in querying. E.g. if we want to retrieve "a scene where is present a man in front of a table", we define the restriction (`TEST-C retrieve-geometric-rel-test MALE TABLE in-front-of`).

It's worth noticing that ICARUS can adapt to different application domains by defining domain-specific knowledge bases, without modifying other system components, but only providing a suitable user interface.

Parsing and Geometrical Analysis

The main purpose of ICARUS is the analysis of three-dimensional scenes modeled using the Virtual Reality Modeling Language (VRML). The VRML file format integrates 3D graphics and multimedia and is widely used in a distributed environment such as the Web.

An important characteristic of VRML files is the ability to compose files together through inclusion and to relate files together through hyperlinking. Hierarchical file inclusion enables the creation of arbitrarily large worlds that can be dynamically modified through a variety of mechanisms that encourage composition, encapsulation, and extension.

Each VRML file:

– Implicitly establishes a world coordinate space for all objects defined in the file, as well as all objects included by the file;
– Explicitly defines and composes a set of 3D and multimedia objects;
– Can specify hyperlinks to other files and applications;
– Can define object behaviors.

In a VRML file it is implicitly defined a scene graph, which contains hierarchically grouped nodes describing objects and their properties, as well as nodes that participate in the event generation and routing mechanism.

Prototypes allow the set of VRML node types to be extended by the user. Prototype definitions can be included in the file in which they are used or defined externally. ICARUS provides its users with a rich collection of prototypes, i.e. 3D-object models defined in terms of other VRML nodes, which can be used to model a virtual scene.

The system has specific information about its prototype library, such as their possible use (furniture or characters), the concept that represents them and their appearance (original dimension, orientation and position). This information is used during the analysis process in order to obtain a CLASSIC description of a scene. VRML offers different mechanisms in order to include instances of prototypes in a scene:

▪ Direct use of EXTERNPROTO nodes, i.e. prototypes defined externally;
▪ Definition and reuse of PROTO nodes, i.e. prototypes node defined completely in the file, including others prototypes;
▪ Use of INLINE node, in order to include directly another VRML file;
▪ Definition and reuse of grouping nodes that include others prototypes.

Parsing a VRML file, the ICARUS Analyzer is able to identify all the instances of the system prototype library that are present in a virtual scene. At the same time absolute position in the world coordinate space is calculated for each identified instance. For this purpose, the analyzer uses both local geometric transformations (scaling, translation, and rotation) extracted from the scene graph and initial geometric knowledge of prototypes.

The parsing phase is followed by a geometrical analysis, whose purpose is to retrieve special geometric relationships between recognized prototype instances. In particular the relations that can be determined between two objects are:

- *Symmetrical relations*, which join a pair of objects.
  - Near: both dimensions and relative distances between objects are used. A special use of this relation involves determining objects that are near to viewpoints in the scene;
  - Opposite: involving objects position, dimension and orientation. Objects must have inverse orientations.
  - Behind: as for opposite, but the same orientation is required.
  - Inside (Outside): used to describe if an object is completely contained in another one.
- *Asymmetrical relations*, which relate an object to another one, but the inverse relation is not necessarily true. Each relation involves dimensions and positions of both objects, while orientation is implicitly assumed choosing a specific face of the object.
  - In front of;
  - Back;
  - Beside;
  - Over;
  - Under.

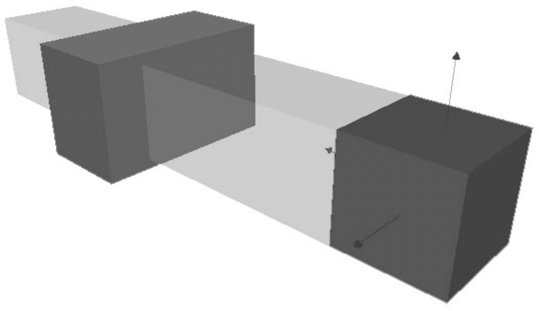

Figure 3. Relation between bounding-boxes

Geometrical analysis is carried out taking into consideration not objects themselves but their bounding-boxes, i.e. rectangular parallelepipeds surrounding objects.

For example, the relation "in front of" between the object bounding-boxes is shown in figure 3. Note that before starting the analysis of the relations the bounding-boxes of the objects are scaled (in function of their average dimension) and their projection planes inclined. Using this technique we can get better results in computing the relationships, because we can take into consideration both the dimensions and the distances between objects.

The last part of the analysis process aims at constructing a description of the analyzed scene in the format of CLASSIC. In particular, the analyzer creates individuals with specific properties:

– An individual for each prototype instance;
– An individual for the scene.

Each individual is defined indicating the most significant concept it satisfies, such as `TABLE`, `BED`, for furniture prototypes and `MAN` or `WOMAN` for character prototypes. Positions and orientation information are stored and others individuals are indicated as geometric role fillers, according to geometrical relationships retrieved. Colors can be related to `ACTOR` individuals through specific color roles. The individual related to the scene is defined indicating that it satisfies directly only the concept `SCENE`. All others interesting concepts are deduced during classification. A scene is described by listing all the objects that are contained in it. Moreover information about defined viewpoints is also stored.

An important phase in defining a description is *role-closure*. In fact CLASSIC adopts the *Open World Assumption* and only if roles are explicitly closed each individual can be well classified taking into account all its properties.

Querying the KB

ICARUS provides its users with the capability of submitting their VRML files for analysis. If no error is reported during parsing phase, the system creates new individuals and classifies scene description in the KB.

The other main service provided by ICARUS is querying the KB and retrieving information about scenes. For this purpose a suitable graphic interface is available.

The user can obtain a scene description, in the form of a list of all defined-concepts the scene satisfies. Selected scenes can be visualized and the user can navigate through the virtual worlds.

Using a query panel one can specify compound queries, in order to retrieve all scenes:

– Whose environment is a particular room (kitchen, sitting room, library, bedroom) or a garden;
– Where specific characters (Alice, Bob, …) are necessarily present (or not present);
– Where there is at least, at most or exactly a certain number of elements. An element can be a specific kind of object (Chair) or a particular actor (Frank), but generic categories of element (Furniture or Characters) are also allowed.
– Where a specific geometric relation between elements (objects or actors) is satisfied. E.g. Bob near a bed.
– Where all (or at least two) characters are in a particular relationship. E.g. a scene with all friend or at least 2 lovers.
– Where there is at least a character with specific clothes or physical features (blue trousers or white beard).
– That can be classified as particular situation, such as a job meeting, a romantic scene, a Christmas party, etc.

An important feature is incoherent queries detection. Then error messages are shown in dialog frames. Inconsistent queries cause only warning messages.

Examples

We propose some examples to show how ICARUS can be used. Notice that we used only prototypes present in the system library to model these scenes.

Example 1. This scene is visualized in figure 4. The analysis of this scene recognizes the presence of three characters and several pieces of furniture. An interesting defined-concept the environment satisfies is `KITCKEN`. In fact the scene is a `ROOM` because the prototype of a room was used, and all the furniture that are indicated in the `KITCKEN` concept definition (a cooker, a sink, a fridge, a table, two chairs, a cabinet) appear. This scene can be retrieved by a query that requires, for example:

- Exactly three characters;
- At least two female characters;
- A female actor near a table;
- A generic character in front of a chair;
- A male character opposite a sink;
- A fridge beside a cooker.

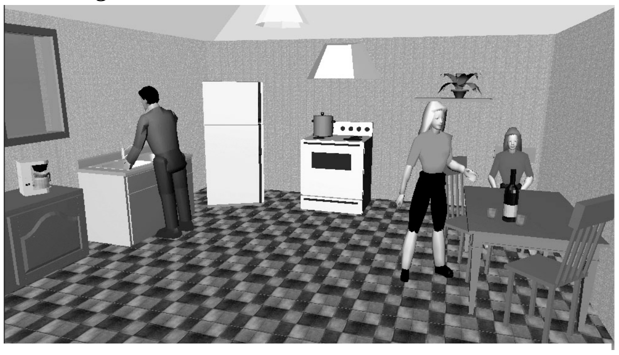

Figure 3. Example 1

Example 2. A view of the scene in this example is shown in figure 5. Note that this is a `SCENE-IN-GARDEN`. However the scene is classified as a much more specific concept: a `ROMANTIC-SCENE`. To understand why it can happen you must know that the two actors represent the characters `ALICE` and `FRANK`, who are known as lovers. Hence, all conditions that describe our idea of romantic scene are satisfied: there are exactly two lover characters and they stand near and opposite.

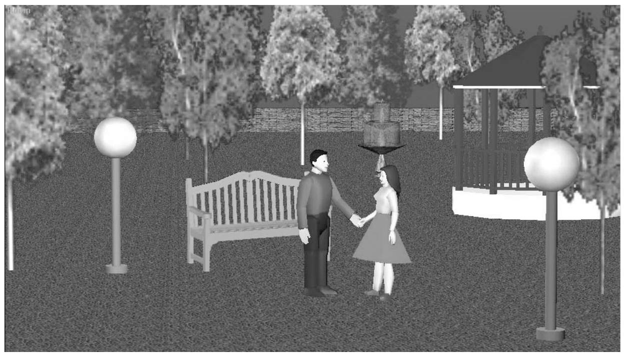

Figure 5. Example 2

Conclusions

In this paper we have presented a prototype systems that provides domain specific content-based classification and retrieval of VRML scenes. In our system the emphasis is on the representation of the logical and geometrical properties of scenes and their use for classification and retrieval. Many other systems aim at classifying complex data (e.g. images) by the use of logical relations. For example in [4] Corridoni et al. extract from an image a set of relations based on the color and these relations are then used for retrieval. While we are not aware of any other system that use a concept description language, such as CLASSIC, for the classification of 3D scenes, it is worth mentioning the system ALFRESCO [5] that uses YAK (a concept description language) for accessing information on 14th century Italian frescoes and monuments.

An important research issue that needs to be addressed in the near future is extending ICARUS analysis mechanisms to other 3D file format (CAD format). Another fundamental aspect to investigate is description and classification of dynamic scenes. Since this behavior is expressed via an event model, we are currently working on a logical language that can express the dynamic relation between objects.

References

1. Adjeroh, D. A.; and Nwosu, K. C. 1997. Multimedia database management: Requirements and issues. *IEEE Multimedia* 4(3): 24-33.
2. Brachman, R. J.; McGuinness, D.L.; Patel-Schneider, P. F.; Alperin Resnick, L.; and Borgida, A. 1991. Living with CLASSIC: when and how to use a KL-ONE-like language. In Sowa, J. F., editor, *Principles of Semantic Networks*, 401-456. Morgan Kaufmann, Los Altos.
3. Calvanese, D.; De Giacomo, G.; and Lenzerini, M. 1996. Structured objects: Modeling and reasoning. In *Proceedings of DOOD-95,* 229-246. Number 1013 in LNCS.
4. Corridoni, J. M.; Del Bimbo, A.; De Magistris, S.; and Vicario, E. 1996. A visual language for color-based painting retrieval. In *Proceeding of IEEE Symposium of Visual Languages*, 68-75.
5. Stock, O. et al. 1993. Alfresco: enjoying the combination of natural language processing and hypermedia for information exploration. In Maybury, M. T., editor, *Intelligent Multimedia Interfaces,* 197-224.
6. Nwosu, K. C.; Thuraisingham, B.; and Berra, P.B. 1997. Multimedia database systems: A new frontier. *IEEE Multimedia,* 4(3):21-23.
7. Pazandak, P.; and Srivastava, J. 1997. Evaluating object DBMSs for multimedia. *IEEE Multimedia,* 4(3):34-49.
8. Subrahmanian, V. S. 1998. *Principles of Multimedia Database Systems.* Morgan Kaufmann, Los Altos.

Acknowledgments

We thank Francesco M. Donini and Emilio Domenicucci for their work on a previous version of ICARUS. Work partially supported by ASI (Italian Space Agency) and MURST.