Location-Aware System Based on a Dynamic 3D Model to Help in Live Broadcasting of Sport Events *

Roberto Yus
University of Zaragoza
Maria de Luna 1
Zaragoza, Spain
ryus@unizar.es

Eduardo Mena
University of Zaragoza
Maria de Luna 1
Zaragoza, Spain
emena@unizar.es

Jorge Bernad
University of Zaragoza
Maria de Luna 1
Zaragoza, Spain
jbernad@unizar.es

Sergio Ilarri
University of Zaragoza
Maria de Luna 1
Zaragoza, Spain
silarri@unizar.es

Arantza Illarramendi
Basque Country University
Manuel de Lardizabal s/n
San Sebastián, Spain
a.illarramendi@ehu.es

ABSTRACT

Broadcasting sport events in live is a challenging task because obtaining the best views requires taking into account many dynamic factors, such as: the location and movement of interesting objects, all the views provided by cameras in the scenario (some of them wireless, mobile, or attached to moving objects), possible occlusions, etc. Therefore, a technical director needs to manage a great amount of continuously changing information to quickly select the camera whose view should be broadcasted.

In this paper, we present a location-aware system that helps technical directors in the broadcasting task, using a 3D model updated continuously with real-time location data retrieved from the scenario. They can indicate in run-time their interest in certain moving objects and the system is in charge of selecting the cameras that provide the kind of views required.

Categories and Subject Descriptors

H.2.4 [**Database Management**]: Systems – Query processing; H.3.4 [**Information Storage and Retrieval**]: Systems and Software; H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems

General Terms

Algorithms, Management

Keywords

Content selection in run-time, 3D model, multiple camera management

*Area chair: Gang Hua

MM'11, November 28–December 1, 2011, Scottsdale, Arizona, USA.

1. INTRODUCTION

Broadcasting organizations are considering the combined use of professional cameras with low-cost cameras controlled remotely to reduce the production cost. However, that solution has a serious impact in one of the most complex and critical tasks in a live content production environment: the more cameras are employed, the more images are available, and therefore the more complicated the work of the technical directors is. Technical directors have to make quick decisions in live broadcasting scenarios to select, among the available video sources, the best video stream to broadcast.

In this paper, we present a system that, using 3D models as basis, assists technical directors in the difficult task of selecting in live the best candidate video signals coming from static or moving cameras. Here we improve the ideas proposed in [4] by considering 3D scenarios, and so we tackle problems like occlusions and computing the percentage of an object being viewed by a camera. Thus, the system supports an expressive specification of the kind of view that must be obtained (e.g., a side view showing at least a 60% of the object, etc.). So, we could highlight the following benefits provided by the proposed system:

- It manages information about the features of video cameras to select those that could satisfy the requirements of the technical director (the kind of views needed on certain target objects).
- It considers the 3D representation and orientation of the objects, in addition to the features of the cameras (focus, pan, tilt, etc.), to recreate the camera views (taking occlusions into account).
- It ranks the cameras presented to the technical director. For example, the percentage of the required object that each camera can provide can be selected as a ranking criterion.

The rest of the paper is structured as follows. In Section 2, we explain how objects in a scenario are modeled. In Section 3, we describe the approach proposed to obtain the cameras that can provide the specified views, based on different 3D computations. In Section 4, we present an experiment that shows the efficiency of the system in terms

of processing time. In Section 5, we review some related works. Finally, conclusions and future work are included in Section 6.

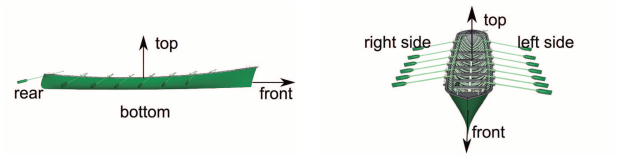
2. MODELING THE OBJECTS

To model objects in the scenario, a generic class *Object* is defined, which represents the set of entities in the scenario. An *Object* is represented by the tuple:

<id, name, loc, extent, centroid, frontV, topV, cams>

where *id* is a unique identifier of the object, *name* is the name of the object, *loc* is the location of the object (its 3D coordinates), *extent* is the volume occupied by the object, *centroid* is the centroid of the object extent, *frontV* and *topV* are normal vectors pointing towards the front and the top of the object, respectively[1], and *cams* is the list of cameras attached to the object (if any).

In order to define the extent of the object, a 3D mesh created by using an external modeling package (e.g., Autodesk 3ds Max, Blender, etc.) should be made available to the system. Besides, the normal front and top vectors must be defined for this 3D model in order to allow the system to distinguish between the different kinds of views of the object (top/bottom, front/rear, left/right, or any combination of two or three elements chosen from the three previous pairs). In this way, the system will be able to answer queries retrieving, for example, cameras focusing a certain object, cameras recording a top view of an object, etc. Figure 1 shows the 3D model that has been created to represent the rowing boats of the use case explained in Section 4, their normal vectors, and some example views that we could obtain.



(a) Side view (b) Front/top view

(c) Top view (d) Rear/right/top view

Figure 1: Different views of an object

3. 3D SELECTION OF CAMERA VIEWS

In this section, we explain how the system, based on the use of a 3D model, computes the answers to the queries requesting camera views on certain target objects. First, we explain how the percentage of the target object viewed by a camera is computed. Then, we explain how the system computes the kind of view (front/rear, top/bottom, right/left) of an object that a camera is providing.

[1]The current orientation of the object is provided by a compass and a 3D accelerometer.

3.1 Percentage Viewed of the Target Object

In [4], the keywords *full* or *incomplete* can be specified as a parameter for a query in order to indicate that a complete or any view of the target object must be obtained. However, this simple approach has some limitations. On the one hand, a *full* view means that the camera is focusing the whole object. Therefore, even if only a small percentage of the object is not in the view, the corresponding camera will be excluded from the answer. On the other hand, if the technical director asks for an *incomplete* view, cameras focusing a very small percentage of the object will be also part of the answer.

Our new proposal solves these limitations by supporting queries that ask for a certain minimum percentage[2] of an object or a certain minimum percentage of a specific view of the target object (defined by the corresponding vectors, as explained in Section 2).

In order to compute what a camera is viewing, we recreate the scene in a 3D engine by taking into account the camera features (current focus, pan, tilt, etc.) and the 3D models of the objects. Then, we use 2D projections (obtained by rendering the 3D scene) to calculate the percentage of an object that is viewed by that camera. For this purpose, we have used *JMonkeyEngine* (JME)[3], a Java game engine with advanced graphics capabilities supported through *OpenGL 2.0* via *Lightweight Java Game Library* (LWJGL). We update the location and direction of the objects and cameras in the 3D scenario continuously every second, to obtain accurate results. Since the system has less than one second to perform all the calculations, an efficient approach has been developed to obtain the answers as soon as possible, which is evaluated in Section 4.

There exist two situations where a camera could have an incomplete view of an object: 1) when the target is partially or fully occluded by another object, and 2) when the target does not fit the field-of-view (FOV) of the camera. In the following, we summarize the computation of the percentage that a given camera is viewing of a certain target object (illustrated by using the scene of Figure 2(a)):

1. The scene is recreated in the 3D engine using the location, direction, and 3D models of objects viewed by the camera that we are considering.

2. The target object is painted completely in red, including the parts outside the FOV of the camera, and the other objects in the scenario are painted in transparent green, in order not to hide the target object (see Figure 2(b)).

3. The current FOV is painted in transparent blue to select what the camera is currently viewing (see Figure 2(c)).

4. If the target does not fit completely the FOV, the virtual camera in the 3D model is moved backwards until it views the target object completely. This movement allows the system to obtain a render covering the full object while it does not affect the perspective of the scene (see Figure 2(d)).

[2]We believe that specifying a maximum percentage is not very useful, but it could be added easily to the system, if needed.

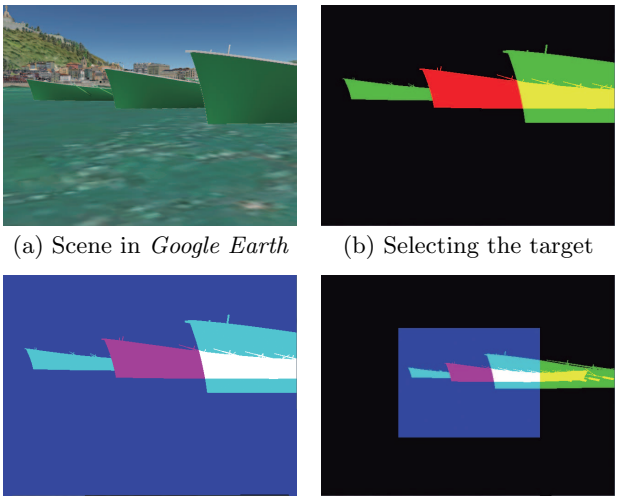
[3]`http://www.jmonkeyengine.org/`

5. A 2D projection of the 3D scene is rendered, and then the system obtains the percentage of the target object viewed by the camera[4]:

$$\frac{\#pixels\ not\ occluded}{\#pixels\ of\ target\ object}$$

Using the image of Figure 2(d), the system obtains that the camera views 41% of the target object (the second boat). Notice that the visible part of the target object is shown in magenta because of its red and blue channels and the occluded part inside and outside the FOV are shown in white and yellow, respectively.

Due to the use of different color channels, the system is able to manage occlusions in different situations, and the above computations are performed with a single render and iteration.



(a) Scene in *Google Earth* (b) Selecting the target

(c) Painting the FOV (d) Full view of target object

Figure 2: Sample computation of the percentage of a target object in a view

The percentage of the target object viewed by each camera can be used to rank the answer presented to the technical director (cameras viewing a higher percentage of the target object are presented first).

3.2 Queries About Specific Views of an Object

A technical director could be interested in obtaining cameras focusing a target object but retrieving a certain kind of view. For example, he/she could be interested in obtaining the cameras that are providing a rear/top/right view of a rowing boat (as seen in Figure 1(d)). To do that, the system needs to obtain the parts of the object belonging to the views requested. As the 3D model of the objects can be complex, selecting the parts of the target that belong to a view can be a difficult and time-consuming task. So, we have used an approach based on *light sources* and *illumination*. The idea is to illuminate only the wanted view by using a directional light source, and thus making it only visible. Besides, by using different colors for each light source, the system can simultaneously check several views in a single pass.

Figure 3 shows an example of this technique, when a technical director queries the system to obtain cameras retrieving a right/top view of a target object. A light source (red) has been set to "select" the parts of the object that belong to the top view and another light source (green) "selects" the parts that belong to the right side view. With this single image, and analyzing the color channels of the pixels, the view of the target object provided by this camera is classified as a predominantly top/right side view (the camera is viewing a 78% of the top and a 92% of the right of the object), and so is returned to the technical director as part of the answer set.

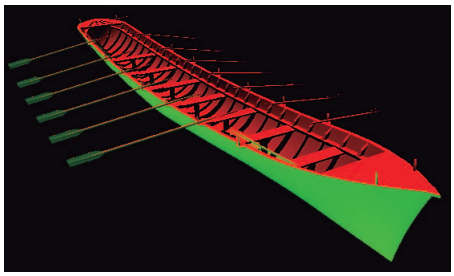


Figure 3: Sample computation of the kind of view of a camera shot

4. EXPERIMENTAL EVALUATION

For the experimental evaluation we have considered a specific sports scenario: the traditional rowing races in San Sebastian, Spain. The scenario has four rowing boats equipped with a *Pan-Tilt* camera and moving according to real GPS location data captured during the race in 2010. Besides, there are other three fixed cameras placed in usual positions for TV cameras. Let us suppose that we want to retrieve the cameras that can view at least 70% of a certain boat. This query will be evaluated continuously, and so the answer will be always kept up-to-date. The proposal is based on the location-dependent query processing system LOQO-MOTION [3], which enables the continuous query processing needed to automatically update (as necessary) the set of ideal camera selections.

In our experiment, we want to evaluate the processing time needed by the system to compute the views of the seven cameras in the scenario. As the location, direction, and other parameters of the objects and cameras in the scenario are updated every second, the system has to be able to perform the processing in less than one second to provide the technical director with updated information.

In Figure 4 we represent the time needed by the system to compute the views of all the cameras (vertical axis) for every time instant during the race (horizontal axis). The tests were run on an Intel Core i5-480M with graphics card NVIDIA GeForce GT 540M. As our approach is based on the use of a 3D engine, the processing time depends on the graphics card (the graphics card used is in the mid-range). Notice in Figure 4 that the maximum time needed by the system was 0.17 seconds. The highest delays are reached when most of the cameras do not focus the target boat and their rotation has to be computed to obtain what the cam-

[4] *pixels of target object*: red channel $\neq 0$
non-occluded: red & blue channels $\neq 0$, green channel $= 0$

eras will see. The minimum processing time needed was 0.078 seconds, very close to the average time (0.085 seconds). Therefore, this test shows that the system achieves the processing time requirements needed for a dynamic environment as the one explained before.

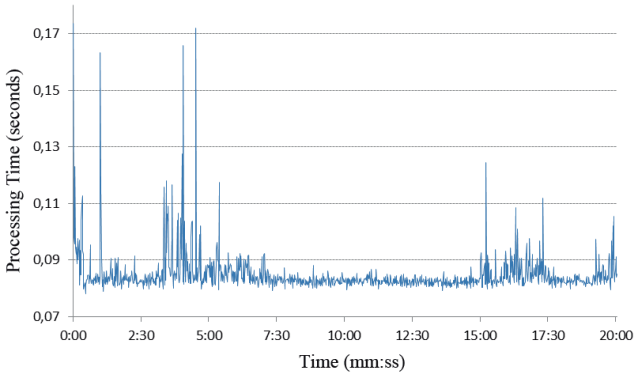


Figure 4: Processing time

5. RELATED WORK

A number of works have focused on the specific problem of helping producers of sport videos. For example, as part of the *APIDIS* (*Autonomous Production of Images based on Distributed and Intelligent Sensing*) project, in [1] the authors present a system that helps the video production and video summarization in the context of basketball games. The work presented in [2] tackles the problem of summarizing videos of soccer games by applying different image processing algorithms to analyze the input videos. In [6] the problem of video summarization, based on metadata describing the semantic content of MPEG-7 videos, is considered in the context of baseball games. Cricket and soccer videos are used to validate the work in [5], which exploits audio features (such as an increase in the audio level of the voice of the commentators or the cheers of the audience) to extract excitement clips from sport videos.

All these works perform an automatic video processing in an offline way, while our approach is based on real-time processing; although some works explicitly mention the possibility of live broadcasting [7], they do no consider moving cameras. Besides, they use image processing techniques (which could fail, for example, for the detection of specific objects located far or partially hidden by others), while our approach is based on a 3D model updated continuously with the current locations and directions of objects and cameras in the scenario. So, our proposal can complement the above approaches.

6. CONCLUSIONS AND FUTURE WORK

In this paper, we have presented a system that helps technical directors in the difficult task of broadcasting sport events in live. The system makes use of a 3D model of the objects in the scenario. This model is continuously refreshed (every second) by taking into account the locations, and other parameters, of the moving objects and cameras. The technical director can query the system in order to obtain the cameras that fulfill his/her demands. The main features of this proposal are:

- 3D models of the objects and camera features (focus, pan, tilt, etc.) are considered to recreate the scenario. In this way, we do not need to analyze real camera sequences. Indeed, our approach can be a good complement for image processing techniques.
- The approach supports an expressive way to indicate the kind of view that must be obtained. More specifically, it supports the definition of the viewpoint required (front/rear, top/bottom, side) and the minimum percentage of coverage of the target object that must be provided.
- The cameras presented to the technical director can be ranked, for example, according to the percentage of the required target object that can be provided.

Moreover, the proposal is quite flexible, and so new functionalities could be added to the system without relevant changes to the main architecture. Thus, we plan to adapt the system to other sports scenarios such as classic cycling. Besides, we will also consider situations where exploiting multimedia information provided by mobile devices carried by spectators or tourists would be interesting.

7. ACKNOWLEDGMENTS

This research work has been supported by the CICYT project TIN2010-21387-C02. We also thank Francisco J. Serón for his technical support.

8. REFERENCES

[1] F. Chen and C. D. Vleeschouwer. Personalized production of basketball videos from multi-sensored data under limited display resolution. *Computer Vision and Image Understanding*, 114(6):667–680, June 2010.

[2] A. Ekin, A. M. Tekalp, and R. Mehrotra. Automatic soccer video analysis and summarization. *IEEE Transactions on Image Processing*, 12(7):796–807, July 2003.

[3] S. Ilarri, E. Mena, and A. Illarramendi. Location-dependent queries in mobile contexts: Distributed processing using mobile agents. *IEEE Transactions on Mobile Computing*, 5(8):1029–1043, August 2006.

[4] S. Ilarri, E. Mena, A. Illarramendi, and G. Marcos. A location-aware system for monitoring sport events. In *Eight International Conference on Advances in Mobile Computing & Multimedia (MoMM 2010)*, pages 305–312. ACM Press, Austrian Computer Society (OCG), November 2010.

[5] M. H. Kolekar. Bayesian belief network based broadcast sports video indexing. *Multimedia Tools and Applications*, 2011. Online First, published online: 9 July 2010, doi:10.1007/s11042-010-0544-9.

[6] N. Nitta, Y. Takahashi, and N. Babaguchi. Automatic personalized video abstraction for sports videos using metadata. *Multimedia Tools and Applications*, 41(1):1–25, January 2009.

[7] J. Wang, C. Xu, E. Chng, H. Lu, and Q. Tian. Automatic composition of broadcast sports video. *Multimedia Systems*, 14(4):179–193, September 2008.