RFID-based Object Detection During Video Production

Tiago Machado
Federal University of Juiz de Fora
Rua Jose Lourenco Kelmer, S/n
Juiz de Fora, Brazil
tiago.machado@ice.ufjf.br

Marcelo F. Moreno
Federal University of Juiz de Fora
Rua Jose Lourenco Kelmer, S/n
Juiz de Fora, Brazil
moreno@ice.ufjf.br

ABSTRACT

Recently, Digital TV Broadcasters have been concerned on how to adapt their workflow in order to embrace the production and delivery of interactive content. Due to the involved changes and costs, broadcasters could not yet fully explore the possibilities of interactivity that would enable an enhanced, pleasing user experience. It becomes essential for broadcasters to rely on new technologies that can lower the costs and assist the production of interactive content with a level of automation as high as possible. In this context, this work proposes an object detection method to be used during video capturing (offline or real-time) that delivers this information as a structured Object Timeline, which can be easily handled for the automated creation of interactive content and other applications. In order to lower the cost of object detection and making it faster and precise, this work evaluates the use of Radio Frequency Identification (RFID) as the relying tehnology for the proposed method.

Keywords

Object detection, RFID, Object Timeline, Multimedia Metadata

1. INTRODUCTION

Digital TV (DTV) refers to a high quality and advanced broadcasting technology of moving pictures and sound, that is more efficient than the analog TV. One of its most promising aspects is that the viewing becomes an enhanced experience, since DTV delivers high-quality audio/video and, mainly, it is a highly interactive technology. However, this interactivity feature brings many challenges on how to make it effective, useful and lucrative.

Indeed, Interactive DTV (iDTV), allows potential consumers to directly interact with new content in the television. The concept introduced by iDTV aims to make the TV more interesting to use than the old TV, providing more information and claiming for the attention of the user on it. It also brings to content providers and broadcasters a new horizon for making business, especially for advertisements.

However, the creation of enagging interactive content is not easy. Broadcasters must be concerned on how to create and deliver iDTV content focusing on a pleasant, enhanced viewer experience. They must find new working methods that makes feasible the creation of iDTV content on a day-by-day basis. This means that these working methods must provide some degree of automation. The more automated and integrated into the video producing workflow such a method is, the lower will be the complexity and cost for creating iDTV content. However, achieving a high degree of automation is not trivial, specially for the creation of interactive content that is directly related to the broadcasted video. By closely relating interactivity and video scenes, broadcasters will be able to deliver an engaging experience to the viewers.

In this scenario, acquiring comprehensive information about a scene is essential. Specifically, if information about relevant objects present in a scene is known, iDTV content can be created to allow viewers to interact with those objects. Obviously this kind of multimedia metadata is useful for many applications besides iDTV. Other examples include recommendation systems, audio/video search and retrieval systems and audio description generators.

This work proposes a robust solution named TimedObjects that allows for the automated generation of information about detected objects[1] during video shooting in recording studios or even on real-time transmissions. Throughout the course of detection, the system generates an editable Object Timeline that depicts the permanence of the objects in scene and may export this information to other applications.

Detecting objects in video is generally a time consuming and expensive task. In order to simplify the object detection, lower its cost and making it faster and precise, this work evaluates the instantiation of the architecture using a mature technology of object identification, the Radio Frequency Identification (RFID). The proposed proof of con-

[1]Objects for this work uses a broader definition. The use of the word must be extended to animated or inanimate characters. The main idea is to identify objects in scene as well as people, scenario, background color, landscape, i.e., anything relevant.

In: Workshop Internacional de Sincronismo das Coisas (WSoT), 1., 2016, Teresina. Anais do XXII Simpósio Brasileiro de Sistemas Multimídia e Web. Porto Alegre: Sociedade Brasileira de Computação, 2016. v. 2.
ISBN: 978-85-7669-332-1

cept embraces all the stages in the object detection chain, starting with registering the objects in the system, capturing them, and exporting the Object Timeline to an editing tool. Although it is not part of this proposal to have a unique destination to the Object Timeline, a use case targeted at the creation of interactive content is also presented.

The reminder of this paper is organized as follows: Section 2 discusses related work; Section 3 presents the TimedObjects system architecture. Section 4 describes the Object Timeline structure. Section 5 demonstrates how RFID equipment has been used as a basis for the RF-TimedObjects system. Section 6 presents experimental results of the RF-TimedObjects and Section 7 is reserved for final remarks.

2. RELATED WORK

Despite the use of object detection term being widely used in computer vision field, it is also largely used in systems based on Radio-Frequency IDentification (RFID) [3]. RFID is a technology for automated identification of objects and people. RFID is a manner of storing and retrieving data through electromagnetic transmission to an RF-compatible integrated circuit. The system is mainly composed of two components, the RFID reader and the transponders (tags). Its main usage is to tag an object and, using the reader, identify if the tag is in the reader range or not. Due to its simplified operation, RFID has shown a low complexity and powerful object identification system.

The RFID tags can be active or passive. Active tags generally have an attached battery that amplifies the signal and the reading range. The passive tags usually have a minor reading range. Generally, the RFID tags are previously indexed to an object, then, the task to detect an object is summarized in reading the tag.

As observed in [2], RFID is considered a real-time object detection technology, i.e. a computer system where the correctness of its behavior depends not only on the logical results of computations, but also on physical time when these results are produced.

A general method for embedding information concerning items appearing in video using RFID tags is proposed in [1]. The patent describes generically an invention in video production field. The main idea is to automatically embed information concerning items appearing interactive video by using RFID tags. Since the work is a patent, the focus is to describe the idea of the invention, not considering important steps on it. Although the work describes the use of RFID tags to identify objects in video production and its association in interactive content, it does not specify main patterns necessary in the process, such as the format of the metadata associated with the video, the method used to synchronize the metadata and the captured video and all the technology used in the interactive content.

A large number of related work resides on the computer vision field, where video or images content is analysed in order to detect features that helps to form the object hypothesis. Computer vision techniques may be applied for object detection during video production. However, the complexity of computer vision algorithms, the processing delay and the required hardware make that kind of solution expensive if compared to our RFID-based solution. Morover the fast detection provided by RFID solutions makes our solution more suitable for live content applications.

3. TIMEDOBJECTS ARCHITECTURE

This section details the functional blocks that are essential for system operation. The working method and interaction between them are also presented. Figure 1 illustrates these functional blocks.

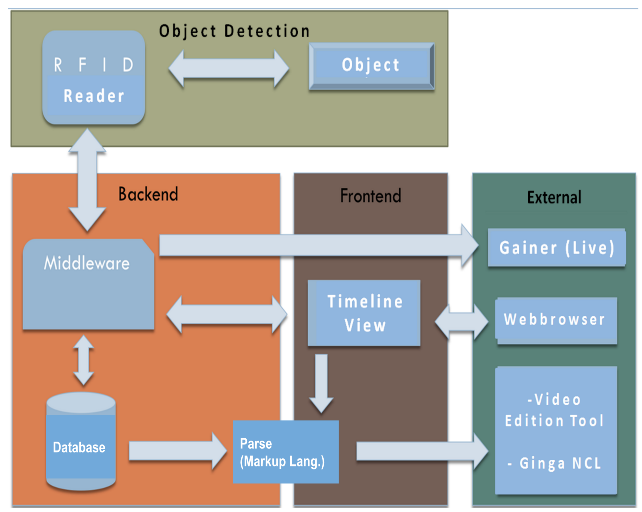

Figure 1: Functional blocks general intercation.

As the name implies, the Front End block is responsible to detect the object in the scene while it is being recorded. Basically, the Object Detector must provide to the system the information of the identified object such as its name or identification code and the moment (time) when the identification has occurred.

As a functional block, the Object Detector does not depend on the technology used. At this point, it is only important to describe the functionality of the block, no matter how it works.

The Back End can be considered the core of the system. Primarily it works as a middleware to the Object Detector, i.e., it is in the Back End that the detecting parameters are set. Main attributes such as period interval time and duration of the detection are assignments of the Back End.

Once the middleware is set, the Object Detector will provide the information of the read objects in a chronological order. It is in the Back End that the Object Timeline is structured: it processes the detection information and create the timeline based on previous information provided by the Object Detector block. As the detector only sends the detected objects, the Back End manages all the scene situations, processing if the object is new or old in scene and if it was in or out of the scene in the last read. In this way, the duration of the object in scene is calculated.

To handle with the objects information it is important to register the objects. The Back End must have access to a database that allows the user to register the tags and informations such as object description and object images. Therefore, the system will have to deal with two kinds of objects: the registered and non-registered ones. The timeline will only be created based on the registered objects. Hence, if at any time the system detects an object in scene that is not registered, it will treat this information, present it to the system operator, but will not place the object in the timeline.

Finally, the Back End is responsible to export the Object Timeline in raw format to be used in other systems or functional blocks. Being XML (eXtensible Markup Language) a widely used standard for data exchange between systems, it has been adopted in this work.

It is worth mentioning that the system control should be made by the Back End. Even though other blocks can have similar functionalities, every request should pass through the Back End. For example, the start and stop of the object detection and, consequently, the creation of the timeline are all concentrated in the Back End.

The Front End functional block has two major purposes. The main one is to provide to the user a better visualization of the timeline. This module is responsible to communicate to the Back End and organize the information in a easy way to be presented to the final user. The Front End provides the user interface where system operator interacts with the Object Timeline. System operators may interact with the presented timeline and modify it as needed. The operations over the timeline are described in Section 4.

After the interaction with the timeline, there will be two different timelines. The first one is the timeliine as registered in the Back End, named raw timeline. As a consequence of interaction by the system operator, the Front End should process the timeline and allows it to be exported as an XML file. At this point, the system will have two different timelines for the same scene, the edited one in the Front End and the raw one in the Back End. Both timelines may be persisted for future use.

The last functional block, the External Elements, represents all other systems that receive an Object Timeline to a specific purpose or, secondary technologies that may assist system operators but are not essential to the main operation of the system. The Web browser used to visualize the Object Timeline is an example of an External Element.

As we propose to export Object Timelines, there may exist various External Elements that will not be depicted in this work. Specifically, there are two envisaged purposes for External Elements that we would like to highlight.

First of them is the capability of Video Editing tool to process Object Timelines as a metadata of the video and use it at editing time. Another application includes technologies and systems that can handle the structured Object Timeline to generate automated content in iDTV environment.

In both cases, the system must provide a well-strucutred Object Timeline. It is not in the scope of this work to handle how those environments will process Object Timelines.

The system can also provide a real time stream of the detected objects and their information in a live broadcasting scenario. For this purpose, the Back End sends the information of detected objects directly to a functional block called Gainer. In this scenario the workflow could skip the Front End functional block. The Gainer must receive the object detection strings and process them as intended.

4. OBJECT TIMELINE

The so-called Object Timeline must represent the permanence of one or many objects in the scene, i.e., in course of time, from the moment that an object is identified until the moment it is no longer identified. Therefore, the timeline will carry the permanence of each object in scene, not merely the existence of an object in scene. Each object can have its own timeline, but for an easier understanding and displaying, all the objects are arranged in the same timeline in the TimedObjects system.

An object can also have its permanence registered one or more times. In case of an object that never leaves the scene, it will have a continuous register in the timeline. Otherwise, in case of leaving, the register will contain every permanence period of the object in the scene. The timeline will therefore present gaps between every register of the object that separates the periods of permanence.

Timeline creation is based in two parameters: the detection period and the duration of detection. The detection period describes the time difference (interval) between the beginning of a read and the next read. Whereas the duration of detection defines for how long the Object Detection functional block will try to detect objects.

The Object Timeline will be created considering the Object Detector reading period. While an object is recognized, the timeline may be affected in one of three ways: (i) if the object was never detected in the scene, the system will add it; (ii) if the object is already in the timeline, the system updates and increases its permanence time; (iii) if the object was detected before but out of the scene the system will add it again creating a new permanence period.

The presence of an object that is already registered in the timeline is dictated by the detection period. If the object is in the timeline and in the next detection period it is not recognized, it will be considered as no longer in scene. If after that the object is detected in a future reading period, it will be considered as back in scene and thus a new register for it will be included in the timeline. The timeline for a specific object will not present any interruption, indicating that it did not leave the scene, if the object is identified in every detection period.

Since each Object Detector will have its specific behavior, reading intervals and duration, the decision whether an object remains on the scene or not requires a compromise between the timings of the Object Detector and the accuracy of the timeline. For example, with a detection period of 5 seconds and the read duration of 1 second, the detector will be deployed for 1 second in each 5 second interval. Thus, if an object leaves the scene for more than 6 seconds (reading time plus the duration of the reading), in its next appearance it will be registered in the timeline after a gap. If the object goes ou and back to the scene so it is read again in the same 6-second interval, there will be no break in continuity of reading, so the timeline will register their stay in the scene maintaining the continuity. The TimedObjects systems allows system operators to tune these duration and period values so they can adapt the timing according to the characteristics of each shooting.

5. RFID-TIMEDOBJECTS

The RFID-TimedObjects is a proof of concept where the TimedObjects system relies on RFID readers to implement the Object Detectors provided that the relevant objects have been tagged. The Back End presents a user interface so that system operators can set up the system. The Back End is more complex and less intuitive for system operators to set up. All the Front End operation method is also originated from the Back End.

5.1 Back End

The Back End has been developed in Java. The system stores all the registered tags in a Java Database. In the first moment the system must have the objects that will be in scene previously recorded in the database. The tag is registered with normal parameters such as ID, content and images. By default the tag is only passive to the system, which means that all tags only refer to an object. Optionally the tag can operate as a command for the system. This property is set in the register screen and dictates how it will interfere in the system operation.

A Start-tag or End-tag can, in addition to refer to an object, operate as a command for the system. Once the tag is set to one of those parameters or both, the object will act as a trigger to start the Object Timeline or finish it, respectively.

The reading method for the Object Timeline is also set in the Back End. The start of the Object Timeline capture can be set to a user request or to a standby mode. This mode makes the reader active but the timeline will only start its execution if a Start-tag is detected. In this way, as soon as an object registered as Start-tag is identified, the Object Timeline will start.

Independently if controlled by a tag or not, once the timeline is started, it can also be stopped directly by an user request or, as occurs in the case of the Start-tag, the capture can be set to wait for an End-tag. When the End-tag is read, the Object Timeline is stopped.

The reading interval is related to the periods that the RFID reader will be reading, i.e., how often the reader will capture tags. The reading length is related to the period that the reader will actually be reading. In additional to those two parameters, the system also allows the user to enter an error rate. This error rate is set in milliseconds and directly interfere in the Object Timeline creation.

Finally, it is possible to export the list of captured objects directly in the Back End. This functionality preserves the raw information of the Object Timeline, i.e., the righteous record of the registered objects identified in the scene.

During the timeline construction, the Back End receives tags in the order that they are being identified, creating a list of it. At this point, the Back End only has the information of the moment that the tag was in scene and the reader power when that tag was identified. To create the Object Timeline, the system must infer the permanence of an object in scene. Thus, it is necessary to calculate the permanence of the object according to some criteria.

First of all, the system always use the present time to work, considering day, month, year, hour, minutes and seconds. The system calculates this time in a single variable in milliseconds. Based on this, for every identified tag, the system associates its ID with the milliseconds time of its identification. What dictates whether the tag is still in scene or not is the reading period. To confirm if the tag is still in the scene, the system uses the time of the last read tag, summed to the reading interval and to the reading period. The values are compared to the actual read time.

Besides all the parameters that can be set to the RFID reader, eventually the reading event is not precise as expected. Being RFID a radio propagation technology, it is susceptible to interference. In this context the error rate can be used. To certify that the system will calculate the permanence of an object in scene as desired, the error rate variable can be set and used in the permanence equation:

$$\begin{aligned} ActualTag(time) \leqslant LastTag(time) + ReadingPeriod(time) \\ + ReadingTime(ms) + Errorrate(ms) \end{aligned} \tag{1}$$

With the error rate in the equation, the system operator certifies that, even with a delay in the reading step the permanence of an object will be in accordance to his needs.

The value of error rate will vary and depends on the purpose of the timeline. After build the Object Timeline, the system only has the information of the power for the last read of the object in scene. Thereafter, when the timeline is built, the system does not carry the history of the power variation for this tag but only the power for the last identification of the object.

5.2 Front End

The Object Timeline is completely transferred to the Front End, which is responsible for showing it to the system operator.

The Front End is a web based solution, i.e., the system operator must access it from a Web browser. The timeline visualization runs in an Apache server and uses a dynamic, browser-based visualization library. The library is designed to simple, handle large amounts of dynamic data and to enable manipulation of and interaction with the data[2].

The main purpose of the Front End is to present the objects timeline as they are identified. Thus, the objects are already presented with their permanence on the timeline. As the reading is being performed, if the object remains in the scene it is updated and the bar is increased. If the object leaves the scene the formation of the bar is finished.

[2] www.visjs.org

6. EXPERIMENTAL RESULTS

To perform the experiment, a real use situation was created. The simulation was done with the RFID reader positioned in the same angle of the camera, both camera and reader are in the end of a table. The farthest object was placed 1 meter from the reader, a LCD monitor, which was fixed, i.e., was present in the scene from the beginning to the end. The other objects were placed between the reader and the monitor during the schene[4].

Tee objective with the experiment was to perform a test that encompasses all object detection situations. The control of the scene is done by tag control. The start-tag was set to the Clapperboard object and the end-tag to the PS3 Controller. Two objects will be at the scene at all time, the LCD Monitor and the CD Cover. Despite being in scene, the CD Cover is not registered at the Java database, i.e., it should be read but not registered in the timeline. Other objects will enter and leave the scene. Table 1 describe all the objects (name, tag number and tag functionality) used to perform the experiment.

Table 1: Objects used to real simulation.

Object	ID	Functionality	In DB?
Book	8888	Object	Yes
CD	2222	Object	No
Clapperboard	7777	Object and Start Tag	Yes
LCD Monitor	9999	Object	Yes
PS3 Controller	5555	End Tag	Yes
Wallet	3333	Object	Yes

The created scenario and execution order for the described environment and tags is summarized below:

1. The LCD monitor and the CD are in scene;
2. The Clapperboard appears in the scene;
3. The Clapperboard leaves the scene;
4. The Wallet appears in the scene;
5. The Book appears in the scene;
6. The Wallet leaves the scene;
7. The PS3 Controller appears in the scene.

Figure 2 illustrates the real scenario of the test.

For the created scenario, the RFID reader was started to operate in Tag Control mode, i.e., the object detection was started but until the system identifies the Start Tag (the Clapperboard in this case) the timeline does not start. All the objects identified before the Start Tag are not listed in the Object Timeline. As 2 shows, the CD object is not registered in the database, in this case, it should not be in the timeline, even being detected by the reader after the clapperboard (Start tag) appears.

For the described scene and objects detection, Figure 3 shows the final results of the Object Timeline as displayed to the system operatior.

As expected, the LCD Monitor object was identified in the first moment of the capture but its presence in timeline cannot occur before the Start-Tag (Clapperboard). As Table 2 shows, the system identified two occurrences of the LCD

Figure 2: Real environment. Each frame represents the input end output objects in scene.

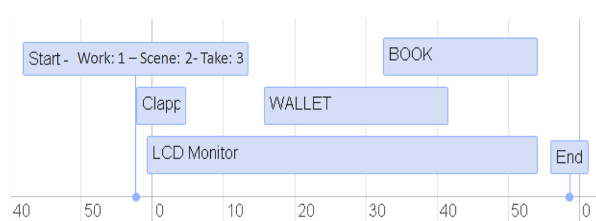

Figure 3: The timeline representing the time that objects appear in scene and its permanence on it.

Monitor before the Clapperboard, but the object was just sent to the Object Timeline after the Clapperboard object, as exemplified in Figure 3. The Wallet object was correctly identified and its presence was marked in timeline until it leaves the scene. The Clapperboard was also registered as an object in scene in addition of been a control tag. The PS3 Controller ended the timeline successfully as an End-Tag. The book appears in the scene and does not leave until the end.

Table 2: Registered object detection

Detect Time	Object
0:02.038	LCD Monitor
0:07.508	LCD Monitor
0:11.484	Clapperboard
0:12.981	LCD Monitor
0:13.021	Clapperboard
0:18.519	Clapperboard
0:18.965	LCD Monitor
0:24.008	LCD Monitor
0:29.396	WALLET
0:30.388	LCD Monitor
0:30.456	WALLET
0:36.126	WALLET
0:36.666	LCD Monitor
0:42.572	WALLET
0:42.680	LCD Monitor
0:46.219	BOOK
0:48.723	WALLET
0:48.783	BOOK
0:49.284	LCD Monitor
0:55.302	WALLET
0:55.833	BOOK
0:55.903	LCD Monitor
1:01.361	BOOK
1:01.892	LCD Monitor
1:07.880	BOOK
1:07.951	LCD Monitor
1:12.828	PS3 Controller

The real experiment and system performance behave as expected. The results have shown that, as a proof of concept, the system worked as predicted.

6.1 Automated creation of iDTV content

In the field of interactive DTV, Nested Context Language (NCL)[3] is a declarative authoring language for hypermedia documents. NCL is an XML application language that is an extension of XHTML, with XML elements and attributes specified by a modular approach [5].

NCL was initially designed for the Web environment, but a major application of NCL is its use as the declarative language of the ISDB-Tb (International Standard for Digital Broadcasting) middleware (named Ginga[4]). It is used to present interactive, multimedia applications [6].

In order to exemplify the use of an External Element for the Object Timeline, we develop an NCL generator based on a T-Commerce (television commerce) template and the main experiment files. The XML generated in the full object detection and the video recorded [4].

The items only appear in the TV for the viewer as they are detected in scene. Figure 4 exemplifies all the object detections and the interaction of the viewer in the iDTV application. The experiment simulates a wallet purchase by the user. The product is added to the cart and the buyer can then checkout. At this point the wallet is out of the scene, so the product is not offered again.

[3]www.ncl.org.br

[4]www.ginga.org.br

Figure 4: NCL example - Each frame represents the viewer interacting with the T-Commerce app to purchase the wallet

To produce this experiment the Object Timeline XML was mapped into the NCL editor with video anchors. Based on these anchors, Ginga-NCL triggers the exhibition or the removal of an item from the list of products available to purchase with the use of links. By mapping the TimedObjects XML to NCL anchors, the system could perform an automated t-commerce application from the Object Timeline

The NCL example shows that the use of automated content can easily be done based on the XML file for the Object Timeline.

7. CONCLUSIONS

In this work, we have addressed an automated object detection system to be used during video production, specially in Digital TV context. Our intention was to provide a robust platform to broadcasters and content producers. We envisage the proposed Object Timeline to be used by video editing software and iDTV authoring tools.

Based on the proposed TimedObject system, we developped our apparatus based on the RFID technology. First a RFID Object Detector was used to perform object identification in scene. Once the Object Detector was selected, we developped the functional blocks to create the Object Timeline and generate an XML file with the description of the permanence of all the objects in scene and also the possibility to broadcast the detected objects for live video stream.

The experimental results have shown that the main goal of the work was achieved. The system could identify objects in scene while it was been recorded, organize it based on the appearance time of the objects and export the permanence of an object to External Elements by means of an XML file.

We also performed the validation of the XML file, developing a generator of interactive Digital TV applications using the NCL language. Based on a T-Commerce template, the generator imports the XML file and, in the moment that the object appears in scene, the product is offered to the viewer. Using the iDTV remote controller and its colored buttons (red, green, yellow and blue), the viewer can buy the offered products.

The results have also shown that, despite all the problems to control the RFID reader, we could use this technology as proposed. The RFID reader has presented a limitation of simultaneous tag reads, it could not identify all the tags in scene if it exceeds five. As the proposed example of an NCL application uses 4 objects, this limitation did not detracted the results.

8. REFERENCES

[1] S. M. Abraham, V. A. Bijlani, and M. Thomas. Method, medium, and system for automatically embedding information concerning items appearing in video using rfid tags, Apr. 26 2011. US Patent 7,933,809.

[2] H. Kopetz. *Real-time systems: design principles for distributed embedded applications*. Springer Science & Business Media, 2011.

[3] J. Landt. The history of rfid. *Potentials, IEEE*, 24(4):8–11, 2005.

[4] T. Machado. Full experiment video. *youtube.com/watch?v=wewQ1UwRKTE*, 2015.

[5] L. F. Soares, M. F. Moreno, and C. De Salles Soares Neto. Ginga-ncl: declarative middleware for multimedia iptv services. *Communications Magazine, IEEE*, 48(6):74–81, 2010.

[6] G. L. d. Souza Filho, L. E. C. Leite, and C. E. C. F. Batista. Ginga-j: The procedural middleware for the brazilian digital tv system. *Journal of the Brazilian Computer Society*, 12(4):47–56, 2007.