

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Tiago Machado

**Automated Object Detection During Video
Production**

Juiz de Fora

2015

UNIVERSIDADE FEDERAL DE JUIZ DE FORA
INSTITUTO DE CIÊNCIAS EXATAS
PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Tiago Machado

Automated Object Detection During Video Production

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Orientador: Marcelo Ferreira Moreno

Juiz de Fora

2015

Ficha catalográfica elaborada através do programa de geração automática da Biblioteca Universitária da UFJF, com os dados fornecidos pelo(a) autor(a)

MACHADO, TIAGO.
Automated Object Detection During Video Production / TIAGO
MACHADO. -- 2015.
78 f. : il.

Orientador: Marcelo Ferreira Moreno
Dissertação (mestrado acadêmico) - Universidade Federal de Juiz de Fora, Instituto de Ciências Exatas. Programa de Pós-Graduação em Ciência da Computação, 2015.

1. Object detection. 2. Video production. 3. Objects timeline. 4. Interactivity. I. Moreno, Marcelo Ferreira, orient. II. Título.

Tiago Machado

Automated Object Detection During Video Production

Dissertação apresentada ao Programa de Pós-Graduação em Ciência da Computação, do Instituto de Ciências Exatas da Universidade Federal de Juiz de Fora como requisito parcial para obtenção do título de Mestre em Ciência da Computação.

Aprovada em 4 de Setembro de 2015.

BANCA EXAMINADORA

Prof. D.Sc. Marcelo Ferreira Moreno - Orientador
Universidade Federal de Juiz de Fora

Prof. D.Sc. Celso Alberto Saibel Santos
Universidade Federal do Espírito Santo

Prof. D.Sc. Eduardo Barrére
Universidade Federal de Juiz de Fora

*I dedicate this work to my
parents, with a very special
feeling of gratitude for their
support and for always encourage
the pursuit of knowledge.*

*A special thank to my
family and friends.*

*"Don't let your
dreams be dreams."
(Jack Johnson)*

RESUMO

Recentemente, tem crescido a preocupação das emissoras de TV em como adaptar seus fluxos de trabalho de modo a produzir e entregar conteúdo interativo de qualidade no ambiente de TV Digital. Devido às mudanças e custos necessários para a implantação da nova tecnologia, as emissoras acabam por não explorar plenamente as possibilidades de interatividade. Assim, para que o usuário tenha acesso a uma experiência avançada e agradável, é necessário otimizar a produção de conteúdo interativo e diminuir seus custos. Para isto, novas soluções capazes de prover uma maior automação na produção de conteúdo interativo devem ser investigadas. Neste contexto, este trabalho propõe um sistema para detecção de objetos durante a produção de cenas de vídeo, capaz de capturar objetos e associar a captura a metadados relacionados durante filmagens, em estúdio de gravação ou mesmo em eventos ao vivo. A informação capturada é entregue de forma estruturada, na chamada linha do tempo de objetos, a qual pode ser manipulada durante todos os estágios de edição do vídeo, bem como na criação de conteúdo interativo.

Palavras-chave: Detecção de Objetos. Produção de Video. Linha do tempo de objetos. Interatividade.

ABSTRACT

Recently, TV Broadcasters have been concerned on how to adapt their workflow in order to produce and deliver high quality interactive content in the Digital TV environment. Due to the changes and costs needed for the deployment of this new technology, broadcasters do not fully explore the possibilities of interactivity. Therefore, in order to provide users an enhanced and pleasing user experience, they are required to optimize the production of interactive content and lower its costs. For this reason, new solutions able to provide a higher level of automation for interactive content production must be investigated. In this context, this work proposes an object detection method able to capture objects and related metadata during video shooting in a recording studio or even on a live event. Captured information is delivered in a structured manner, the so-called object timeline, which may be handled throughout the stages of video editing as well as during the creation of interactive content.

Keywords: Object Detection. Video Production. Objects timeline. Interactivity.

LIST OF FIGURES

3.1	The Apple Final Cut Pro X editing tool	30
3.2	The Adobe Premiere Pro CC	30
3.3	Use proposal for the object timeline on iMovie tool	31
4.1	Functional blocks in high level approach	33
5.1	Inside an RFID card - Antenna, rfid chip, common magnetic stripe and com- mon credit card chip.	41
5.2	A coin size RFID keyring with antenna and micro-chip inside.	41
5.3	back end screen in Java	44
5.4	Tag register screen	45
5.5	Raspberry Pi Architecture	50
5.6	Object timeline example for Objects 1, 2 and 3.	51
5.7	Object timeline - remove selection property	51
5.8	Object timeline: object 3 removal	51
5.9	Object timeline - objects 1 and 2 edited	52
5.10	Back end XML file (left column) and front end XML file (right column). . . .	52
5.11	Implemented technologies used to RFID-based AODVP	53
6.1	Object detection setting with 3 tags.	56
6.2	Object detection setting with 4 tags.	56
6.3	Object detection setting with 5 tags.	56
6.4	Real environment.	61
6.5	Final timeline for the real usage experiment.	63
6.6	XML output file from the back end (left side) and front end (right side). . . .	64
6.7	NCL example, LED monitor offer	68
6.8	NCL example, LED monitor and wallet offer.	68
6.9	NCL example, LED monitor, book and wallet offer.	68
6.10	NCL example, LED monitor, book and wallet offer. Wallet added to cart. . . .	69
6.11	NCL example, LED monitor and book offer. Checkout enable.	69
6.12	Anchors markers from the object timeline XML.	69

6.13 Link exhibition for the wallet object.	70
6.14 Link removal for the wallet object.	70

LIST OF TABLES

2.1	Total pixels for each kind of video resolution	22
5.1	RFID frequency band and applications	42
5.2	Power variation execution example	47
6.1	Object detection: 3 objects in 6 reading periods	57
6.2	Object detection: 4 objects in 6 reading periods	58
6.3	Object detection: 5 objects in 6 reading periods	59
6.4	Objects used to real simulation.	60
6.5	Registered object detection	62
6.6	Unregistered objects detection	63
6.7	Reader distance from power variation	66

LIST OF ACRONYMS

- AODVP Automated Object Detection During Video Production
- CZC Crescent Zero Crescent
- CZCy Crescent Zero Cycle
- DTV Digital Television
- DZD Decrescent Zero Decrescent
- DZCy Decrescent Zero Cycle
- GPS Global Positioning System
- HTML HyperText Markup Language
- iDTV Interactive Digital Television
- IEEE Institute of Electrical and Electronics Engineers
- IPTV Internet Protocol television
- ISDB-TB Integrated Services Digital Broadcasting, Terrestrial, Brazilian version
- ISO International Organization for Standardization
- ITU International Telegraph Union
- ITU-T ITU Telecommunication Standardization Sector
- JSON JavaScript Object Notation
- MPEG Moving Picture Experts Group
- NCL Nested Context Language
- PC Personal Computer
- PHP PHP: Hypertext Preprocessor
- PVR Personal Video Recorder

RAM Random Access Memory

RFID Radio-Frequency Identification

SIFT Scale-invariant feature transform

TCP Transmission Control Protocol

TV Television

UHF Ultra High Frequency

USB Universal Serial Bus

XHTML eXtensible Hypertext Markup Language

XML eXtensible Markup Language

CONTENTS

1	INTRODUCTION	13
1.1	MOTIVATION	14
1.2	PROBLEM DEFINITION	15
2	RELATED WORK	18
2.1	OBJECT DETECTION IN COMPUTER VISION	18
2.2	RFID AND OBJECT DETECTION	23
2.3	METADATA AND TIMELINE SYNCHRONIZATION	26
3	ENVISAGED APPLICATIONS	28
4	ARCHITECTURE FOR AODVP	33
4.1	FUNCTIONAL BLOCKS	33
4.2	THE OBJECT TIMELINE	36
4.2.1	Specification.....	36
4.2.2	Timeline Maintenance	38
4.2.3	Timeline Alignment	39
5	DESIGN AND IMPLEMENTATION OF AN RFID-BASED AODVP	40
5.1	THE FULL-FLEDGED PROTOTYPE	43
5.2	THE LOW-COST PROTOTYPE	49
5.3	OBJECT TIMELINE AND FRONT END	50
6	EXPERIMENTAL RESULTS	54
6.1	OBJECT DETECTION	55
6.2	FULL OBJECT DETECTION	57
6.3	ADVANCED FEATURES	65
6.4	OBJECT TIMELINE DELIVERING WITH NCL	66
7	CONCLUSION	71
	REFERENCES	73

1 INTRODUCTION

The world is going through the digital television (DTV) era. Although initial research in DTV has begun in the 1990s, its official implementation only started by the 2000s in most countries. 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 would become 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.

There are many differences in digital and analog signal, but most of them are usually summarized into the image quality aspect. Indeed, analog TV suffers from ghosting images, noise from weak signals, and other interferences that can degrade the quality of the image and sound. But in the other hand, digital TV requires that the audio and video must be digitally synchronized, so reception of the digital signal must be very nearly complete. In practice, even with weak signal in analog TV it is possible to intelligibly watch TV channels, whereas in Digital TV, neither audio or video would be usable.

Essentially, a DTV service comprises the transmission of audio and video digitally processed and multiplexed. The signal is broadcasted through established delivery networks, for example, cable, satellite, terrestrial or optical fibers. The user only needs a set-top box or a TV set (a terminal device) able to receive, demultiplex and decode the digital stream. The main advantages of DTV are: the increased number of transmitted channels, better quality of image and sound and the addition of interactive services.

The number of digital TV households across 80 major countries around the world have hit 675 million by the end of 2011, 1 billion at 138 countries by the end of 2014 (MURRAY, 2006). This means that almost a sixth of the earth population is now receiving DTV signal on their houses. Moreover, also according to Murray (2006), half the world's TV households now receive digital signals. Digital TV penetration climbed from 23.5 percent at end-2007 to 48.5 percent by end-2011. In the same year, China became the largest DTV household nation, with 289 million digital TV homes.

This scenario has led the television into a new paradigm, the Interactive DTV (iDTV),

meaning that the potential consumer will directly interact with new content in the television. A variety of interactive contents and services might be included in DTV. Since the beginning of iDTV, new technologies are emerging to make this kind of content more efficient for broadcasters to explore, so they can be created every day and thus make iDTV more powerful.

Interactivity with TV-related content may alter how people watch TV in the near future. Examples include getting more real time information about what is on the TV such as weather, sports, movies, news, or any other content. iDTV can be considered the main example of the convergence of the entertainment, publishing, advertising, telecommunication and computing. Cauberghe and Pelsmacker (2006) describes iDTV as a new viewing experience that enables an on-demand, participatory, non-linear, two-way communication platform, in contrast to a broadcast, passive, linear and simple entertainment old television.

The concept introduced by iDTV aims to make the TV more interesting to use than the old one, providing more information and claiming for the attention of the user on it. On the other hand, it also brings to the content providers a new horizon for making business, especially for advertisements.

1.1 MOTIVATION

The novelties introduced by DTV brings challenges to both parts of the process, the viewer and the broadcaster. From the viewer's point of view, he needs to be concerned with purchasing a terminal device, besides learning how to use its new functions and available services. On the other hand, broadcasters have to adapt their content producing chain, from the video cameras in the recording studios to the antennas in the transmission towers.

Besides the costs for adapting the video production chain, the new technologies may also bring threats to content producers and broadcasters. Since DTV makes possible the transmission of more channels, it becomes a challenge to keep the user loyalty and the number of users reached per channel by advertisers tends to reduce. Moreover, some equipment allows users to skip the commercials, by using Personal Video Recorder (PVR) function, which records DTV programs. Evidently, some subterfuges can be used to force the user to watch those commercials.

In any event, it is clear that 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 broadcaster workflow such a method is, the lower will be the complexity, cost and time spent 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 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. This work proposes solutions to provide a robust and trustable platform to the broadcaster, targeting the so-called Automated Object Detection in Video Production (AODVP) and delivering this information to video editing software and iDTV authoring tools, in an easy way to be handled. With such a trustable platform, broadcasters will have more resources to smartly generate interactive content that might result in greater user loyalty.

1.2 PROBLEM DEFINITION

The main goal of this work is to define an architecture for the detection of objects in scene during video shooting in recording studios or even on real-time transmissions. Besides the functional blocks for object detection, the architecture specifies a structured object timeline that gathers information about the relevant objects¹ in scene.

The AODVP architecture can be instantiated by relying on different technologies for object detection, as for example Computer Vision methods. However, that approach might not be promising for the context of this work. Its processing cost and detection accuracy are not satisfactory for real-time scenes, apart from the complexity of algorithms and hardware involved.

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

¹Objects for this work uses a broader definition. The use of the word must be extended to animated or inanimate characters. The mainly idea is to tag objects in scene but also people, scenario, background color, landscape, hair style, i.e., anything.

of object identification, the Radio Frequency Identification (RFID). Basically it works with two parts. When an RFID tag passes through the field of the scanning antenna (transceiver), it transmits the information on its microchip (an ID, for example) to be identified by the scanning antenna.

Once the objects have been captured by the RFID reader during video shooting, an object timeline is built synchronized with the scene, taking the form of metadata associated with the audio and video, with markers of when an object was detected in scene and for how long it stays there.

The metadata should also be prepared to be used by video editing software and interactive content authoring tools. Moreover, it is also interesting to make this object timeline editable at the moment of shooting, in this way, an object can be hidden or added to the timeline according to the producer's interest.

It is not in the scope of this work to investigate how a specific video editing/authoring tool will use the object timeline to create content, even though this work will describe how to take full advantage of the object timeline. The focus is limited to the delivery of a structured object timeline that tracks the relevant objects in scene.

In addition to the AODVP architecture and its object timeline, this work also investigates reasonable hardware to be used in each functional block of the architecture. The goal is to integrate hardware that can be used in real-life recording situations. While the scene is being recorded, the hardware will identify the objects and the user (recording director, for example) will interact with the object timeline while its creation.

The proposed proof of concept 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 presented, taking Ginga-NCL (SOARES; FILHO, 2007), the standard middleware of the Brazilian Digital TV System, as a basis.

This work is organized as follows: Chapter 2 discusses related work, considering the state of art of the main purpose of this work. Chapter 3 describes the requirements for object detection during video shooting, the envisaged applications and use cases. The AODVP architecture is presented in Chapter 4, which describes each functional block, the object timeline and its operations. Chapter 5 presents the object detection technology

used in the work, RFID. Chapter 5 also describes both a low-cost and a full-fledged prototypes for the instantiation of AODVP architecture. Finally, the experimental results in a real life usage scenario are discussed in Chapter 6. It also illustrates a simple NCL application that may be generated from an object timeline. Final remarks and the future work are discussed in Chapter 7.

2 RELATED WORK

The amount of visual information such as digital images and videos has become predominant in computer databases. As a result of this increase, researchers for many years have been studying ways of recognize objects in those media formats. In computer technology the term object detection is commonly related to vision and image processing. Due to the advances in this area in the past few years and the diffusion of digital media formats, especially from the Internet, there is a large and promising research field for the area.

Object detection in computer vision and image processing deals with all kind of objects. The main research topics in the area are face detection, object-tracking, object detection and action recognition.

This chapters discusses the state of art in object tracking and detection in video, referring to image processing. It will also describe the state of art of the same subject applied to other technologies such as RFID. Our aim is to establish the foundation of both areas, including a comparison between them. This work will focus on RFID technologies and also real-time object identification with it.

We start by presenting an overview of object detection in computer vision and image processing. Since it is not the scope of this work to scrutinize the image processing field, it will be shown shallowly, for comparison purposes only. Then we present the RFID technology and a singular approach to the main use of it, the object detection. Afterwards will be discussed the use of sensors and technologies such RFID in media production and the implications of the use.

Finally, synchronization concepts will be presented in order to provide basis to the timeline study in this work.

2.1 OBJECT DETECTION IN COMPUTER VISION

In this section we generally describe the state of art in the image processing field and its applications that specifically permeates this work.

Object detection (recognition) and tracking is one of the most important themes in Computer Vision field. In the past decades, the increased research is mainly motivated by its large applications.

The main idea is to identify objects in an image or a video sequence. Usually, humans perform multiple objects recognition effortlessly and almost instantaneously, even if the image vary of viewpoint, shadow, luminosity, scale, size, rotation etc. However, this task is still a challenge for computer vision systems, since the algorithmic description to solve this kind of problem has been very difficult to elucidate. In the last decades, the area has been reasonable implemented an increased.

In this way, as described in Jain et al. (1995), an object recognition system must find objects in real world from an image of the world, using object models which are previously known. Namely, the object recognition problem can be defined as a labelling problem based on models of known objects. The system must recognize a set of labels corresponding to a set of models, already known to the system.

In summary, the object recognition problem is closely tied to the segmentation problem: without at least a partial recognition of objects, segmentation cannot be done, and without segmentation, object recognition is not possible.

In objection recognition tasks, there are four main components to accomplish extraction (JAIN et al., 1995):

- Model database
- Feature detector
- Hypothesizer
- Hypothesis verifier

The image processing workflow starts with the image (video frame), then the feature detector applies operators to images and identifies features that helps in forming the object hypotheses. It is important that the features extracted from the training image be also detectable, even under image changes, such scale, illumination and noise.

There are too many features extractors in the literature. Their use varies on application and final objective.

The main reference in this area of object detection is the SIFT (Scale-Invariant Feature Transform) (LOWE, 2004b) algorithm and its future improvements. The SIFT features are local and based on the appearance of the object at particular interest points. Here-with, it is variant to image scale and rotation and also robust to other changes, such as viewpoint, noise and illumination.

SIFT detect and uses a large number of features from the images, which reduces the errors caused by local variations in image (LOWE, 2004b). The main contribution of SIFT is that its feature descriptor is invariant to uniform calling, orientation and partially of distortion and illumination changes. As any object recognition algorithm, SIFT uses a database of reference images. Once the features were extracted, the object is recognized in a new image, comparing individually each of its features to the database, then it will match the features to find (or not) the candidates. Finally, given the desired accuracy, it will indicate the number of probable false matches. The object that passes all these tests can be identified as correct with high confidence (LOWE, 2004a).

The feature extractor in images depends on the image characteristics. The SIFT algorithm have been subject to a large amount of improvements in the last decade. In Wu et al. (2013), it is possible to see the comparison between the original and other five SIFT algorithm implementations. The results show that each algorithm has its own advantage, which depends on the chosen scenario, but none is dispensable. In that same work, the authors have performed experiments to compare the SIFT algorithm and its variations. The dataset used is formed by medium to low resolution images, i.e., approximately 800 x 640 pixels. The images are either planar scenes or the camera position is fixed during acquisition (MIKOLAJCZYK et al., 2005).

As outlined, the SIFT algorithm uses four main steps, each one consuming considerable processing resources, although the extraction of features to the comparison of probable matches depends mainly on the hardware. The algorithm is usually not used to real-time video extraction, since it depends on database size and, obviously, the hardware power to process it. Other references in the literature usually use low quality images to their experiments and data sets, as described below.

The Viola-Jones (VIOLA; JONES, 2001) object detection framework was developed to achieve real-time and competitive object detection. Although it can be applied to detect classes of objects, it was developed to perform face detection. Even being competitive and real-time, there are a few restrictions on its first use. The face must point towards the camera in image and it should not be tilted to any side.

The face detection is nothing more than an object detection. At this point, it is important to distinguish face detection to face recognition. The framework is only responsible to recognize if an image contains a face or not. It does not determinate to whom the face

belongs to, for example. The algorithm also works with feature extractions to recognize the pattern of a human face.

Since the proposition of those two main works, there was a huge progress in image object detection. In Sigel et al. (2003), an apparatus was proposed to object detection in shooting-time. It exemplifies a system to be used in sports finish line. The camera must be fixed and it will be responsible to determinate if the athlete has crossed the finish line, capturing the start and end moment of the crossing.

Other researches aim to perform object tracking in real-time videos and human action recognition. In a recent work (OH et al., 2012), it was created a real-time moving object recognition processor for high-definition video (720p) video streams. In this work, the described hardware has reached 83% in moving object accuracy.

In (SHOTTON et al., 2013), the idea is to predict quickly and accurately 3D positions of human body from a single image. The works realizes almost real-time predictions with the Kinect gaming platform (ZHANG, 2012). The results showed good results, however, the kind of object is well known and the used hardware is specific to its application.

A real-time and robust tracking system is proposed in (GRABNER et al., 2006). The algorithms runs in real-times environments, combining computable features to do so. The algorithm was tested in 640 x 480 pixels real world images, at 20fps. The work also demonstrates the performance of tracking objects with extremely good performance to low processing cost.

Another crescent research area that uses object detection in video is the "smart" vehicles one. As observed in (GAVRILA; PHILOMIN, 1999), they present a real-time system that can detect people and transit signs to assist the embedded computer system in car to autonomously control it. Despite the good results, the work describes the problems with real-time object detection only using cameras.

In the same field, detection and tracking of moving objects (DATMO) is a central problem. In (MERTZ et al., 2013), the authors developed a system to perform object tracking by lasers to 2D and 3D scenarios. It also detects moving objects, such pedestrians and vehicles with collision alerts. The system works in real-time situations detecting those moving objects. However, the work shows that, despite the good results, it needs improvements in 3D algorithms to decrease their computational cost.

In this subject, one of the most advanced autonomous system for vehicle control is

the one developed by Volvo¹ car brand. As an example, (STEVENS, 2015) the system is composed by lasers, radar, sonar and visual sensing equipment, i.e. it does not uses only cameras to object detection. The system must be absolutely reliable, thus, the objects informations must be precise.

Real-time object recognition is still a challenge in the area, how come, the huge advance in image quality. As known, all algorithms and frameworks extract features from the images and, by this, deals with the image resolution.

In its turn, image resolution area had a huge advance in past few years. The following Table 2.1 exemplifies this advance.

Table 2.1: Total pixels for each kind of video resolution

Video Mode	Frame size (pixels)	Total Pixels
240p	320 x 240	76.800
360p	640 x 360	230.400
720p	1280 x 720	921.600
1080i	1920 x 1080	2.073.600
1080p	1920 x 1080	2.073.600
2000	2048 x 1536	3.145.728
2160	3840 x 2160	8.924.400
2540	4520 x 2540	11.480.800
4000	4096 x 3072	12.358.912
4320	7680 x 4320	33.177.600

As described, the image sets used for the main algorithms rarely exceed 720p resolution. The current image standard for commercial TVs and cameras has reached 4K resolution, i.e., at least 12,582,912 pixels in images. Linearly comparing those resolutions, it is a increase of approximately 13.5 times. This huge amount of pixels is just one more challenge to imaging processing, especially in feature extraction phase.

Furthermore, it is common to create the image data sets in controlled environments. In other words, the scenery is pre-determined, such as camera position, scene luminosity and in scene movements. In datasets with heterogeneous activities, like Youtube videos, the accuracy of the algorithms considerably decreases. In order to improve accuracy, the feature extractors will also need a good knowledge base or, at least, some descriptors that will define the object.

¹www.volvo.com

Another problem in object detection in image processing is the lack of standards, i.e., a practical way to classify the objects from the recognized patterns. A framework for real time object detection and classification was proposed in (FASEL et al., 2005). Despite the work title pretends to deal with all objects, the work focus on face detection and its classifications in the experiments. Even focusing in face classification, i.e., only one kind of object, the results were not good as expected. For example, it could not deal with eye behavior and also encountered difficulties to accomplish robust and accurate classification.

Hence, in image processing field, it is hard to find any work proposing the object recognition of any element with high accuracy in real-time environments, such as live TV broadcasts or the so-called "smart" vehicles subject, both critical systems, considering the response time needed for the detection. The advance in "smart" vehicles researches were only possible by introduction of other technologies, such radars, sonars and sensors, in this case, the image processing by itself is not considered a trustable real-time object detection technology with high accuracy.

2.2 RFID AND OBJECT DETECTION

Despite the use of object detection term is widely used in computer vision field, the term is also largely used in systems based on Radio-Frequency IDentification (RFID) (LANDT, 2005). RFID is a technology for automated identification of objects and people. RFID is a manner of storing and retrieving data through electromagnetic transmission to a RF-compatible integrated circuit. The system is mainly composed by two components, the RFID reader and the transponders (tags).

The RFID reader and tags must be set in a defined radio frequency and protocol to transmit and receive data. Basically, RFID tags are categorized as either active or passive. Active RFID tags operates without power supply, they reflect the RF signal transmitted to them from the reader and add information modulating the reflected signal. In the other hand, passive tags are mainly used to replace the traditional barcode, they only reflect the RF signal with their ID and generally operate in a worst range than active ones (JUELS, 2014).

The popularization of this technology occurs in the 1990s, when it was used in the first open highway electronic tolling system. Since then, it is been also widely used in supply chains and products control in warehouses (LANDT, 2005).

RFID is also a crescent research topic in academic circles. The use of RFID for tracking objects and people, mainly in indoor locations, is increasing. Although GPS is a very powerful device to provide user localization and tracking routes (DEDES; DEMPSTER, 2005), it is not very effective in indoors environments. The use of GPS in indoor facilities, like schools or universities has no effective accuracy to provide the exactly room that a person is located, for example.

The technology is focused on the idea of object detection. The main usage is to tag an object and, using the reader, identify if the tag is in the reader range or not. Since its main usage is focused in this simple application, RFID has shown a low complexity and powerful object identification system.

As observed in (KOPETZ, 2011), the 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.

In contrast, the use of RFID in object detection can be simpler, since it is linked to the use of RFID tags. Generally, the RFID tags are previously indexed to an object, then, the task to detect an object is summarized in reading the tag.

In (YANG et al., 2013), it is described a system to track an object by RFID with high accuracy and precision. The system uses mid-range RFID readers to detect the so called SRE benchmark, which is defined as the ratio of the number of successful RFID tag reading to the total number of RFID tag readings attempted. In this case, good accuracy was only reached in straight line situations.

The majority use of RFID outside academic circles are concentrated in logistic and surveillance systems. In (CHOW et al., 2006), a complete system to automatic data identification with RFID is shown, assisting logistics service providers in resources planning and execution. The work includes a complete framework to control all the shipping process of products, including real time information such, where the resources are and what they are doing, for instance, loading or unloading.

In a different way, (GONZALEZ et al., 2006) proposes a novel model for warehousing focused in analyzing the massive data sets provided by the use of RFID in logistics systems. The work intended to elaborate efficient methods for data-mining the information collected by the use of RFID.

The amount of data provided by those RFID controlled systems, depending on its scale, can be huge. This information can be used in real-time systems or stored to post processing. In this work we considered all the collected information from RFID readers as metadata of the system. The main idea is to use the RFID systems always synchronized with the time of the object (tag) identification event.

The work of Kumar et al. (1998) describes a simple and intuitive framework to metadata visualization. The main visualization for the metadata is the so-called Timed-Based Metadata Visualization, which align the objects in rows and columns. From left to right, the columns describe the increasing time and each new row describes an specific object.

In Moreno and Machado (2013) we propose a portable and low-cost prototype system to identify objects for video production. The system use a close range RFID frequency (13.56Mhz), the user must approximate the reader to the object (less than 3cm), almost touching it. The detection was transmitted to a server by the Raspberry-pi wi-fi device and the server create a chronological timeline registering the moment of the object detection. Although the system was pretended to be a low-cost, in case of a scene with many actors, it is necessary to provide a system for each of them. Besides, the objects that are not very close to the actors are not detected in scene.

A general method for embedding information concerning items appearing in video using RFID tags is proposed in Abraham et al. (2011). 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.

The patent also describes, based on images, the architecture to the proposed system. Basically the system pretend to tag products and identify the location of the products while video is produced, then the video and RFID information are merged in a server where the RFID information is embedded into the video. After that the data-embedded video is made available for viewing by users. Finally, when the video and embedded data

is immediately displayed to user, he can purchase or receive a message with additional information concerning to the item. It becomes clear that in Abraham et al. (2011) the authors intended to register the idea in a patent and not any kind of appliance, since they not describe any fundamental requirements for an elaborate development of the idea such metadata and the object timeline. Thus, it is not possible to reproduce this work. Beside that, the patent describes just two specific applications for the system. At any time the authors relate way of presenting the captured objects.

In Marianantoni et al. (2004) it is proposed a complex use of sensors in a film set to record lighting, equipment used, locations of lights, camera, and actors, as well as light intensity measurements, by fusing data from several different sources and store them into a database. As before, the authors did not describe the metadata used in this project, neither how this information can further be used with other applications.

2.3 METADATA AND TIMELINE SYNCHRONIZATION

The advance in DTV brought new parameters to video patterns. The increase in services and content types brings new parameters to this technology, such metadata. As described in (LUGMAYR et al., 2004), metadata is defined as an additional data used to describe data, or, in a simplistic way, it is also known as "data about data".

According to Alves et al. (2006); Durand et al. (2005), the use of metadata in DTV is focused in three main patterns: MPEG-7, MPEG-21 and TV-Anytime. Accordingly to the work, all of the three formats have shown good results to handle with metadata. In DTV, it becomes challenging to opt to any of them. The use of eXtensible Markup Language (XML) (DACONTA et al., 2003) is an example of successful use of metadata in Internet and other applications. Given its wide dissemination and support for most diverse applications and computer areas, it will be the metadata language adopted in this work.

When dealing with one or more multimedia sources, most of the systems should provide a method to synchronize the different sources into one single content. The most common alignment is between audio and video sources. There is a need for systems and methods that correctly and accurately calibrate the timeline for audio transcripts with the underlying audio and video.

Despite not being in the scope of this work to propose the timeline alignment, the

topic will be introduced and briefly addressed.

In Berry and Yang (2012) it is proposed a method to timeline alignment for closed-caption text using speech recognition transcripts. The presented methods and systems adjust and align the timeline of text within a video by comparing closed captioned words against the text output of detection speech software run against the same video.

According to (STANDARDIZATION, 2013), the term timeline alignment is used in many different contexts. As described, this exploration focuses on the problem of playing ancillary content that is time-aligned to some main content, and probably delivered using a different transport, physical network and/or encapsulated in different containers. The ancillary content may be played on the same device as the main content, or on another device.

3 ENVISAGED APPLICATIONS

This chapter will introduce the main envisaged applications, the use cases pretend on this work and the main requirements for its operation. Some examples of practical usage of the apparatus and the basically functional and nonfunctional requirements will be given. The focus of this chapter is to illustrate the usage scenario but not to present the architecture of the system. The architecture and the explanation of how those next steps occurs in low level will be made in Chapter 4.

As described before, this work pretends to identify all the objects in a specific scene, register when an object enter or leave the scene. With this register, it will generate an object timeline with the permanence of the object in scene. The object timeline could be saved to future usage such as an XML file or, in case of live shootings, streamed.

Some possible applications for this automated content will be addressed. First and the most important one is the usage in Interactive Digital Television, specifically at advertising area: By way of example, consider a recording studio for a talk show program. Usually this kind of program have a main presenter sitting behind a table and, at some point, he calls one renowned person to be interviewed, who comes into the scene. If this person and all the clothes are previously tagged with RFID tags, in the moment that he enters the scene the system will identify all the tags, i.e. all the objects in scene. In possession of this information, the broadcaster can easily create an interactive content to use this objects and offer, for example the clothes of the renowned person to sell or to make any advertise with the brand of it. Outside advertising area, the broadcaster can also generate automated extra content based on the person such person age, works historic or even any gossip about him.

In the example above, the scene is previously known. By this, the broadcaster can product the interactive content based on the scene, but the automation of this process is not simple. Using the scene scripts to automate the interactive content is not a precise method. If the scene is recorded different then predicted, the synchronization of the content to the scene can be compromised. In video production, it is common to accept scenes that are not exactly as predicted.

This is a simple but effective example of usage for this automated content. Obviously

the broadcaster can already generate this kind of advertising without the automated object detection. But automating the object capture the broadcaster can establish templates for advertising to be shown without the interference of an editor.

A second usage for this platform is the so-called in-video research. This field is widely explored by the content providers. For example, the biggest TV channel in Brazil, Rede Globo¹ in its head office at Rio de Janeiro had, by the year of 2009, 322.000 video tapes archived (GLOBO, 2009). The task of search a specific scene in this huge amount of video is impossible without the computer assistance. Much of the video is stored with previously registration of its contents.

If at any time anyone needs a marriage scene at a soap opera, for example, the user will have to search in all tapes this topic. Of course the tapes are already linked to the text of the scene, generally, the script of the scene. Then the user can search by this content to find the image of it. In this way, the research for specifics topics will return all the marriage scenes in soap operas.

Another purpose of use, based on this kind of research is to find a scene by its association with the objects on it, and all this automatic. When stored, the scene will be associated with the metadata information captured at shooting time. By saving the object timeline as a metadata of the video, the search production of content associated with the video will be automatic. By this, the search of any kind of content of the video will be based on the tags. Remembering that this information can be anything that was previously tagged, such objects, actors, scenery, landscapes, etc.

Further this use, the object timeline can carry extra information that can help, to generate information in movie captions, for example. By knowing when an object is or not at the scene, the caption can easily use it to inform the user its situation.

Another application is to correlate the tags to its context. When correlating objects, future and advanced studies in semantic field can be promising. As example, if an ambulance and a gun are detected in scene, the system can infer that someone is hurt. This inference is just based on the detected objects in scene.

The examples above illustrates how the object timeline can be widely use in various scenarios associating the scene with the object timeline on it as its metadata. The use of the object timeline from a video scene can be extended to video editing tools and be

¹www.redeglobo.com



Figure 3.1: The Apple Final Cut Pro X editing tool

easily integrated to them.



Figure 3.2: The Adobe Premiere Pro CC

The video editing tools generally have the same visualization pattern and operation logic. The Figures 3.1 and 3.2 illustrates respectively the interface of the two most power

and famous editing tools, the Apple Final Cut Pro X² and the Adobe Premiere Pro CC³. Both editing tool works with a similar interface for authoring video. In the upright part there is a video visualization to preview the work, in the left part the editing tools. Media such videos and pictures from the computer library presented in the left part. At the bottom part is located the control for the audio and video timelines. In those timelines, the editor can control the flow of a video, audio, pictures, transitions, texts and effects. Usually video archives carries audio associated, in those tools it is also possible to separate video and audio.

The idea of this work is to generate an object timeline also associated with the video and audio archive. By this, it will be possible to integrate the timelines and make those softwares to go beyond the existing timelines. An object timeline can carry the time information to associate in which moment of the scene the object is appearing and also the picture and description of it.

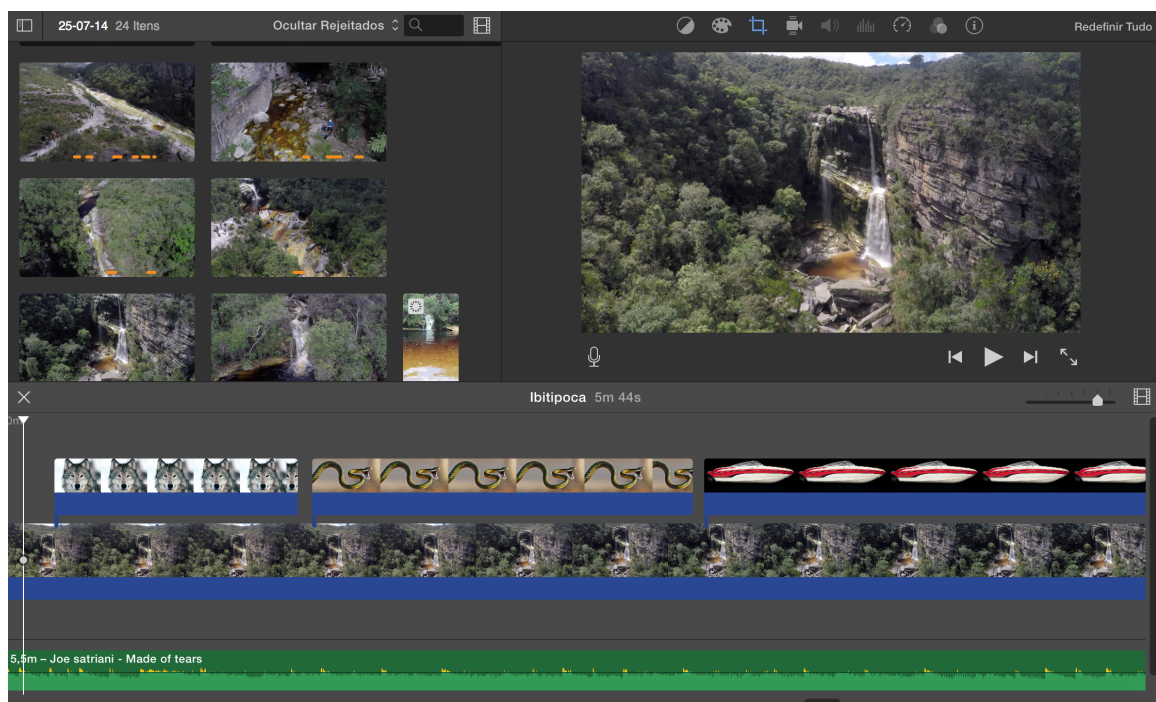


Figure 3.3: Use proposal for the object timeline on iMovie tool

Figure 3.3 illustrate an example of how the editing tool can handle with the object timeline. The software used for this example is the Apple iMovie Software⁴. In this

²<https://www.apple.com/final-cut-pro/>

³<http://www.adobe.com/products/premiere.html>

⁴<https://www.apple.com/mac/imovie/>

example, the software shows 3 timelines. The first timeline represents the object timeline, the three *objects* wolf, snake and boat are represented by the permanence of them in scene. Above the object timeline is the video timeline and the last one is the audio timeline.

As described, this chapter introduce the main envisaged applications to the proposed architecture. Based on the proposed technologies, architecture and also the results of this work, many other technologies can be implemented.

4 ARCHITECTURE FOR AODVP

The architecture for Automated Object Detection During Video Production is detailed in this chapter. The specifications of the functional blocks and their role in the system will be first presented. Therefore, the chapter will describe the features and concepts of the object timeline. It is noteworthy that the architecture presented here is independent of the technology used in this work.

The implementation of the concepts presented in this chapter will be more detailed in Chapter 5.

4.1 FUNCTIONAL BLOCKS

As previously described, this section will detail the functional blocks based on the significance of them in the system. The functional blocks of the system are the parts of the system that are essential to its operation. Thus, it is important to separate them when describing their functionality. The working method and interaction between them will be presented. Figure 4.1 illustrates the functional blocks and its interactions.

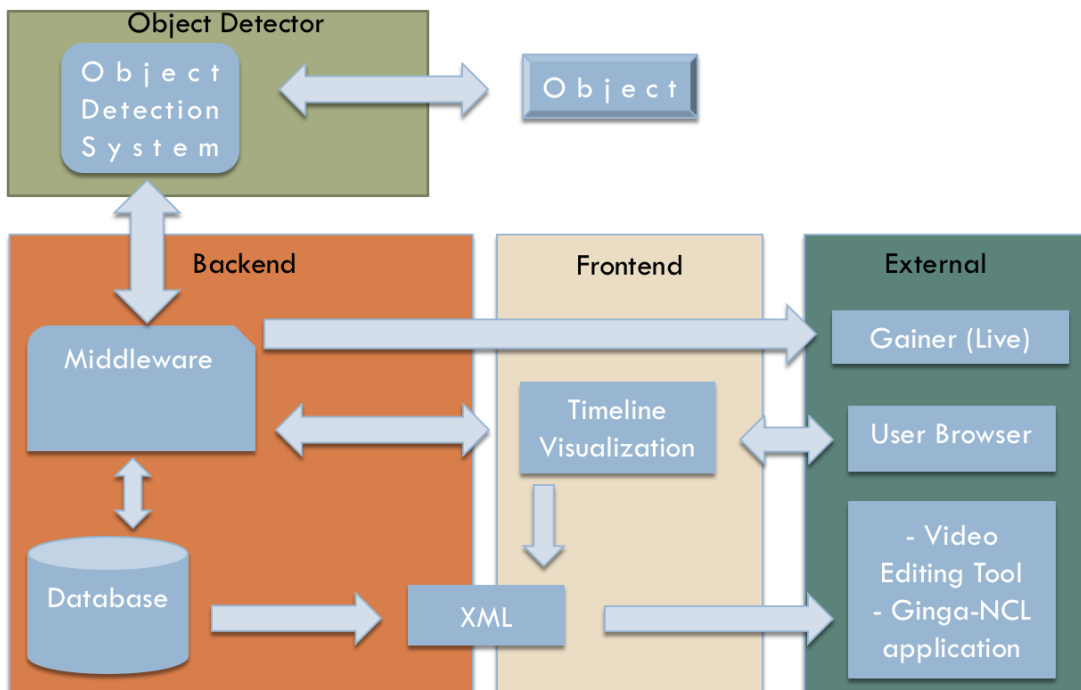


Figure 4.1: Functional blocks in high level approach

As the name implies, the Object Detector block is the responsible to detect the object

in the scene while it is been recorded. Basically, the object detector must provide to the system the information of the identified object such its name or identification code and the moment (time) of the identification.

Generally, the identification systems are controlled by a firmware or middleware. The setup of the system must be a concern to the back-end functional block.

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 work.

The back end is 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 period interval time and duration of the detection are assignments to 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 organized, it also process the information and create the timeline based on previous information provided by the Object Detector block. As the detector only send the detected objects, the back end manage with 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; by this, 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 a database to allow the user to register the tags and informations such object description and object image. Based on this, the system can handle with two kind of objects, the registered and non-registered ones. The timeline will only be created based on the previous registered objects. Hence, if at any time the system detects an object in scene that is not registered, it will treat this information, present to the user but will not be place in the object timeline.

Finally, the back end is the responsible to export the object timeline in raw format to be used in other systems or functional blocks. As described before, been the XML (eXtensible Markup Language) a widely pattern used export and import data between systems, it will be adopted in this work. Other patterns such JSON (CROCKFORD, 2006) can also be used.

It is important to notice that the control of the system should be made by the back

end. Even though other blocks can have similar functionalities, the requisition 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. It is also in the front end where the user interacts with the object timeline. The final user can interact with the presented timeline and modify the original one as needed. The operations in timeline are described in the Object Timeline section.

After the interaction with the timeline, there will be two different timelines. The first one is the one that is registered in the back end, denominated the raw timeline. As a consequence of the user interaction, the front end should process the timeline and also export it as a XML file. In 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.

The last functional block, External Elements, is composed by all the technologies that receive the object timeline to an specific purpose or, secondary technologies that assist the final user but are not essential to the main operation of the system. The user browser used to visualize the object timeline is an example of a secondary technology.

The External Elements are more related to the envisaged applications. As the system proposes to export an object timeline, there may exist a lot of external elements that will not be depicted in this work. There are two main idealized purposes for external elements as follows.

First of then is the capability of Video Editing tool to process the object timeline as a metadata of the video and use it at editing time. Another application is to reach technologies and systems that can handle with the structured object timeline to generate automated content in Digital TV environment.

In both cases, the system must only provide the object timeline structured and up-rightly. It is not in the scope of this work to handle how those environments will process the timeline.

The system can also provide the real time stream of the detected objects and its informations to live broadcasters. For this purpose, the back end must send the information of the detected objects directly to the, so-called, gainer. This workflow could skip the front

end functional block. The gainer must receive the object detection strings and process it as intended.

4.2 THE OBJECT TIMELINE

This section will describe the features based on the object timeline. Herewith, it will not be described the object detection apparatus. Rather the operation of the detector, it will be detailed the framework of an object timeline considering a generic object detector system that deliveries the information of the objects to the timeline. The mentions of the detector in this chapter will consider only its capability to send to the back end the identified object and the time of the identification.

4.2.1 SPECIFICATION

The so-called object timeline must contain the permanency of one or many objects in the scene, i.e., in course of time the moment that an object is identified until the moment it is no longer identified. By this, the timeline will carry the permanency of each object in scene, not merely the existence of an object in scene. Each object can have its own timeline, for ease of understanding and display, all the objects will be arranged in the same timeline.

As previously described, the object can be animate or inanimate and it must have four main attributes: object id, content, start time and end time. Other attributes such image can also be added.

- ID: It identifies the object and controls the duplicity (primary key);
- Content: It describes the object;
- Start time: Register the time of object detection in the timeline;
- End time: Register the time of the object departure of the timeline;

An object can also have its permanency registered one or more times. In case of an object never leave the scene, it will have a continuous register in the timeline. Otherwise, in case of leaving, the register will contain every permanency period of the object in scene.

The timeline will present gaps between every register of the object to separate the periods of permanency.

The register of an object in the timeline and the creation of it is strictly linked to the object detector. The detector should work in specific periods of time, for example, it can work in periods of 5 seconds. Every 5 seconds the detector will send to the object timeline the identified object and the time of its identification. In this case the object can be one or more, depending on the detector capability.

In this way, the object timeline will be created considering the detector reading period. While an object is recognized, the timeline acts in three ways: First, if the object was never detected in the environment, i.e., in the timeline, the system will add it. In the other hand, if the tag is already in the timeline, the system updates and increases its permanency time. Finally, if the tag is out of the environment (timeline) the system will add it again creating a new permanency period.

The presence of an object that is already registered in the timeline is determined by the detection period. If the object is in the timeline and in the next detection period it is not recognized, to the object timeline it will be considered that the object is no longer in scene. After that, if the object is detected in the next reading period, the timeline will consider that the object is back in scene and will create a new entry for it. The timeline for a specific object will not have any interruption, indicating that it did not leave the scene, if it is identified in all detection periods.

In addition to the detection period, the system also has the duration of detection. In other words, the detection period describes the time difference (interval) between the beginning of a read and another. Whereas the duration of detection defines for how long the object detection system will try to discover the object. As a rule, the duration of detection can never be superior to the detection period.

It is also important to define if the time of the duration of detection is *inside* the detection period or not. If the time is *inside* the detection period it will not influence on it, thus, the detection period will work normally. However, if the duration of detection increases the detection period (*outside*), the detection will only iterate after the detector stops the detection, i.e., after the duration of detection. In practice the "inside" period does not influence in the detection period.

To identify whether an object remained on the scene or not, it is necessary to handle

with the time of the detector and the time of the timeline. Since each detector has its specificity such reading intervals and duration. As example, with a detection period of 5 seconds and the duration of 1 second, the detector will take place for 1 second after a 5 second intervals. Thus, if an object leaves the scene for more than 6 seconds (reading time plus the duration of the reading), in its next appearance will again be created in the timeline. If the object in a range of 6 seconds is read again, there will be no break in continuity of reading, so the timeline will register their stay at the scene maintaining the continuity.

4.2.2 TIMELINE MAINTENANCE

In addition to the records automatically captured to the timeline, the objects can also be manually controlled by those tools: create, move, edit and delete.

To perform those controls to the user, the timeline must have two registers, the raw timeline (in back end) and the edited one (in front end). The raw timeline will always keep the non-edited timeline, providing the user the information of the timeline composed exclusively by the detected objects. Therefore, if necessary the user can interact with the timeline, the edited timeline will be the result of the user interaction and recorded by the front end.

Although the automation proposed, any object can be manually insert to the timeline, hereupon the user must provide the name of the object and the time interval of its appearance.

The object edition allows the operator to change the permanence time of the object in the scene and change its description name.

Preserving the object features (name, description and duration) the move function only moves the object to another part of the timeline, i.e., the user must provide the new start time of the object. As the function preserves the object duration, the end time will also be known.

The timeline objects can, according to the user's needs, be stopped and then resumed at any time. Thus, the registration of objects during the stoppage and the timing are interrupted until the resumption in capturing is requested.

In order to provide greater compatibility between systems and ease the import and export of such information, all the timeline information are exchanged by the use of XML

files. In addition, the timeline can be exported in raw or edited format.

4.2.3 TIMELINE ALIGNMENT

Timeline alignment is an old problem in multimedia systems. Normally those systems have to align video, audio and captions. Generally the timeline alignment is automated done based on time and any kind of marker.

The timeline alignment at editing time, i.e., when the object timeline is been used by an editing tool, for example, must be handle by the metadata plugin on the tool. For live streams, it is also belongs to the gainer to deal with the alignment. The scope of this work is to provide the gainer the time of the captured tag, but not its synchronization.

As the object timeline is all generated based on the system time and the objects are registered in the moment of its identification in the scene until it leave, it is important to accurately mark the start and end of the capture. The timeline can be started by the user or by a tag with start or end functionality. In both cases the system marks in the timeline the exact moment of the start and end of the scene.

Another known method to mark the scene and help in the alignment is the use of audio as a marker. The clapperboard is used for years to sync scenes, as it make noise when used at the beginning of a scene record, the editing tool can deal with this specific sound to facilitate the synchronization. An RFID clapperboard can be used as a marker. It can be built just as a common clapperboard with the possibility to activate a tag when clapped. Thus, when it claps, the audio will be captured and the start tag associated to the clapperboard will be activated. This association will auxiliary the sync by the noise (audio) to the object timeline initialization (Start control tag).

5 DESIGN AND IMPLEMENTATION OF AN RFID-BASED AODVP

This chapter will generally describe the RFID architecture and also specify the technology in this work. Initially it will be presented the RFID fundamentals and a brief history. Next the most used frequencies, types of tags and common applications.

Then at section 5.2, a low-cost prototype will be presented, based on a credit card sized computer, the Raspberry-pi¹. The low-cost prototype is a simplification of the Full-fledged prototype, described at section 5.1. Both prototypes share the same main architecture. Finally the Object Timeline based on RFID will be presented and all the expenses of it in the project.

One of the earliest registers RFID was published in 1948, by Harry Sotckman. According to (ROBERTS, 2006), it was a landmark paper called "Communication by Means of Reflected Power". After that, the technology started to advance in the Second World War. Despite the use in the 1950s, the technology had only breakthrough between the 1970 and 1980, where the theory of RFID was developed, some tests and initial implementations started. The popularization and widespread use started in 1990s to present day, after the creation of the standards and the devices price reduction (LANDT, 2005).

RFID is a wireless or non-contact technology for automated identification of objects and people. RFID systems are divided in two parts, an RFID device (tag) is a small microchip designed for wireless data transmission, generally it is attached to an antenna and commonly located on the object to be identified. The second part is the reader or interrogator, which, depending upon the design and the technology used, can be a read or read/write device. Usually, the data capture device is referred to as the reader. Although some readers can write in the tags, changing their default factory ID.

The RFID tags are transponders (acronym to TRANSmitter / resPODNER). Their function is to transmit and respond to the radio-frequency received by the reader. RFID tags are devices encapsulated with an antenna and one micro-chip with information about the tag, for example, an ID. Generally tags are small devices like stickers, cards and keyrings. Figure 5.1 and Figure 5.2 exemplifies RFID components, respectively a common

¹www.raspberrypi.org



Figure 5.1: Inside an RFID card - Antenna, rfid chip, common magnetic stripe and common credit card chip.



Figure 5.2: A coin size RFID keyring with antenna and micro-chip inside.

credit card with RFID antenna and a chip inside, and a RFID keyring and its comparison to a coin in terms of size.

According to Santini (2008), RFID tags can be divided in three groups:

- **Passive:** This kind of tag does not handle a transmitter, i.e., the tag only reflect back the radio-frequency signal from the reader. This kind of tag is the most common, simple, without battery and cheaper;
- **Active or Semi-Active:** Opposed to passive, handle a transmitter and a battery. This kind of tag also send signals to enhance the read range, even with a transmitter it is necessary the use of a reading device. The active tags usually have the battery to power their transmissions, moreover, the semi-active ones use the batteries to power

their circuitry when interrogated by the reader;

- Two-way tag: A specificity of active tags, also handle internal battery. The difference is in the communication method, this kind of tag does not need to be active by a reader, they are capable to start the communication between other tags without the reader (transponder).

Beyond those two main components (reader and tags), RFID devices also include a controller. It vary in complexity, for example, it can be embedded in a smartphone or a dedicated hardware. Usually it carries the middleware responsible to communicate the RFID reader to a computer, for example.

An RFID system generates and radiates electromagnetic waves, they are classified as radio systems. The frequency allocations for radio systems generally are managed through individual countries. There are three main ranges of frequency used for RFID applications. The frequencies also interfere in reading range and reading speed, the Table 5.1 resume the frequency bands and applications in RFID systems.

Table 5.1: RFID frequency band and applications

Frequency Band	Characteristics	Typical Applications
Low 100-500Khz	-Short medium read range -Inexpensive low reading -Speed	-Access control -Animal identification -Library control -Car immobilizer
Intermediate 10 - 15Mhz	-Short medium read range -Potentially inexpensive -Medium read speed	-Access control -Smart cards -Library control
High 850-960Mhz 2.4-5.8GHz	-Long read range -High reading speed -Line of sight required -Expensive	-Railway vehicle monitoring -Toll collection system -Pallet and container track -Vehicle tracking

According to (ROBERTS, 2006), low-frequency passive tags have an effective range up to 30cm, high frequency passive tags up to one meter and UHF passive tags from 3 to 5 meters. To guarantee greater range, some systems use active tags instead of passive, boosting the signal range up to 100 meters.

Generally the time response of an RFID system is less than 100 milliseconds for each read. Some RFID reader can also read many tags instantaneously. Tags can be also read through a variety of visually and environmentally challenging conditions such rain, fog, ice, snow, dust, etc.

One problem encountered in RFID systems is the tag collision. It occurs when there is a lot of chips in the same field. Tag collision occurs when more than one chip reflects back a signal at the same time, confusing the reader. Different vendors have developed different systems to control the tag collision, forcing the reader to identify tags one at a time. Since they can be read in milliseconds, it appears that all the tags are being read simultaneously. The technology used to avoid collision and the limit of simultaneous read are different for each vendor.

The next sections will describe the technologies used to implement the proposed architecture of the system. Two prototypes are proposed in this work. Primarily a simple low-cost prototype will be presented. Next, as an expansion of the first one, a full-fledged prototype is presented with more functionalities and processing power. Finally the object timeline will be presented with its functionalities (front end).

5.1 THE FULL-FLEDGED PROTOTYPE

All the set up in the system is made through a Java application interface, the back end. The back end is more complex and less intuitive to the end user to set up. All the front end operation method is also made in the back end.

Figure 5.3 illustrates the back end screen for the Full-fledged prototype with all parameters.

The first step in the system is to connect the back end to the reader, selecting the connection port.

The system will store all the registered tags in the Java Database. The option to this kind of database instead of other popular ones such MySQL² and PostgreSQL³ was to provide the system a built in database. As the system stores only few informations, the builtin database is a better choice to make the system more simple to deploy in any Java capable system, making it portable. The Java DB is distributed by Oracle⁴ and it does not require any database configuration as a server, for example.

In the first moment the system must have the objects that will be in scene previously recorded. To register an specific tag, it first must be read. The reader operates in two modes, in the tag register situation, it will only use the single-tag mode. This one is

²www.mysql.com

³www.postgresql.org

⁴www.oracle.com

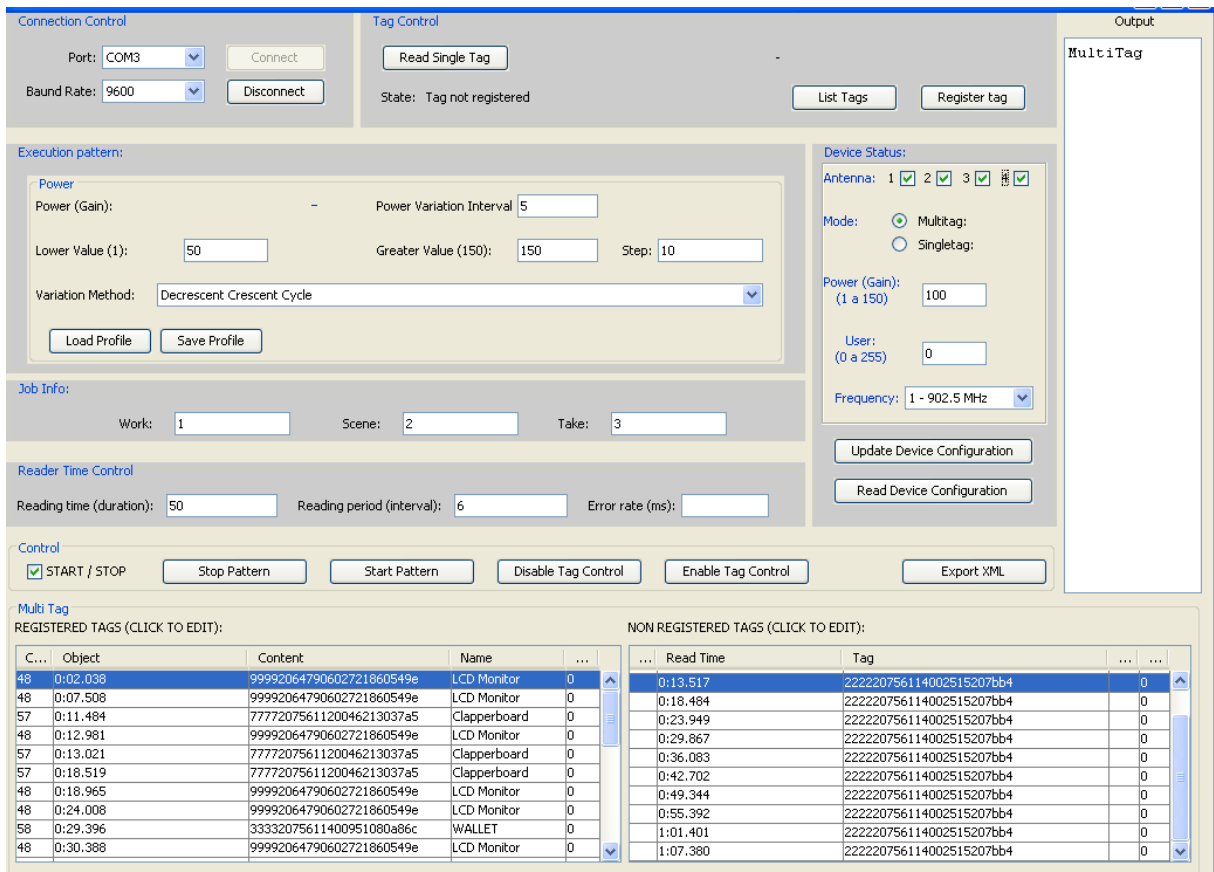


Figure 5.3: back end screen in Java

specific to read one tag at a time, it only activates the reader one single time when requested and the reader send to the back end the identified tag number. If the tag is out of the range of the reader or the reader detects one or more tags, it will not read the tag or will read it with trash information. In this situation the back end discard the reading information and presents to the user the non recognized tag information.

Once the tag is successful read, the operator must register it in the register screen (Figure 5.4). The tag is registered with normal parameters such ID, content and images. As default the tag is only passive to the system, this mean that all tags only refer to an object. Optionally the tag can operate the system, this property is set in the register screen and determinate how it will interfere in the system operation.

When in capture mode, the system always read all the tags in scene, even been registered tags or non-registered ones. The back end make two lists of tags but the object timeline is only created with the registered tags list. Hereupon, the operator can check if there was any tag in the scene that should be registered.

An Start-tag or End-tag can, in addition to refer to an object, operate the system. Once the tag is set to one of those parameters or both, the object will be a trigger to start the object timeline or finish it.

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 make the reader active but the timeline will only start it 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.

Independent to the tag control, once the capture is started, the object timeline can also be stopped directly by an user request or, as occurs in the Start-tag, the capture can be set to wait to an End-tag. When the End-tag is read, the object timeline capture is stopped.

Tag Control:

Tag Control:


Code: Tag:

ID:

Content:

StartTag EndTag

Tag Images



Cod	File	Type
2	monitor.jpg	jpg

Tags List

Name

Cod.	Tag	Name
48	99992064790602721860549e	LCD Monitor
49	1111207561120045233021b9	CD
50	88882064790602690980b36a	BOOK
51	666620647906027619904427	NOTEBOOK
52	444420647906027314108095	SMARTPHONE
54	555520647906027318904c89	PS3 Controller
57	7777207561120046213037a5	Clapperboard
58	33332075611400951080a86c	WALLET

Figure 5.4: Tag register screen

It is also important to set the parameters of the job in the back end, it helps to identify for each take of the scene the timeline belongs to. There are three identifiers: the work (job), the scene and the take.

The Reader State part is specific related to the used reader, thus, it may vary from the reader model. The equipment used in this work has four built in antennas, it is possible to set any combination of them. It is also possible to set the operation frequency from 902.5 MHz to 927.0 MHz. The system has the possibility to set a parameter to identify the user of the system as a number (from 0 to 255).

Last, the 3 main parameters of the reader system are: the power gain - determinate the power of reading distance, reading time (duration) and the reading period (interval).

The majority of the RFID readers available works in two manners. In case of devices with less power, i.e., readers that only read one tag at a time or with less reading distance, they always operate in reading mode. It means that those equipments are all reading and when a tag is close to them, they read it and send the information to the middleware.

On the other hand, readers with more power and consequently more reading distance, usually have the capability to read more than one tag at a time. In this case, those readers generally have two reading parameters to be set, the reading period (interval) and the reading time (length).

The reading interval is related to the periods that the 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. To illustrate how those two parameters work, let consider the reading interval of 5 seconds and the reading length of 1 second. In this way, every 5 seconds, the reader will perform the tag capture for a period of 1 second.

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. This parameter will detailed in the last section, in the object timeline creation logic.

The Execution patterns is an advanced feature to the system. This patterns can be set by the operator to improve the use of the RFID reader and its application. The user can create an execution pattern, save and load it for a future use. The main idea of the Execution Pattern is to use the RFID reader to infer the distance of the object to the reader and also allow the system to focus an object.

The Execution Pattern is related to the power variation of the reader. What is pretended with this function is to infer the reading range of the system and also the distance between the tag and the reader. First the user will set the same reading parameters as

previously described: reading period, reading time and reader power.

After that, five new parameters must be set. The power variation will take place simultaneously to the object identification. First it is necessary to set the power variation range, inputting the lower and the greater values. The step of the power variation is how much the power will vary from the lower to greater value or vice versa. The power variation will also occurs from time to time, and it must be set in the power variation period (interval). Finally it is necessary to define the variation method, there are four preconfigured forms:

- Crescent Zero Crescent (CZC): Vary the power from the lower to the greater value. When it reaches the greater, start to vary again from the lower value;
- Decrescent Zero Decrescent (DZD): Vary the power from the greater to the lower value. When it reaches the lower, start to vary again from the greater value;
- Crescent Zero Cycle (CZCy): Vary the power from the lower to the greater value. When it reaches the greater, start to vary again from the greater value;
- Decrescent Zero Cycle (DZCy): Vary the power from the greater to the lower value. When it reaches the lower, start to vary again from the lower value.

To exemplify the power variation and its attributes, the Table 5.1 will resume all the parameters. For this example the step is set to 10, the power interval to 5 seconds, the lower value is 120, the greater value 150 and the initial power is set to 120. The table resume the operation using the four variation methods in a 35 seconds execution.

Table 5.2: Power variation execution example

Method x Time	0:00	0:05	0:10	0:15	0:20	0:25	0:30	0:35
CZC	130	140	150	120	130	140	150	120
DZD	130	120	150	140	130	120	150	140
CZCy	130	140	150	140	130	120	130	140
DZCy	130	120	130	140	150	140	130	120

Once the tag is detected in a Execution Pattern, it must carry the power value of the reader when it was detected. It is also possible to save the execution pattern for future usage as an execution profile.

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 execution, the back end only receives the tag in order that they are been identified, creating a list of it. At this point, the back end only have 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 process 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. As default, many programming languages use the milliseconds time based on the UTC (Universal Coordinated TIME). This time is a number representing the number of milliseconds between the date object and midnight of 1st January of 1970.

Based on this, for every identified tag, the system associate its ID with the milliseconds time of its identification. What determines whether the tag still in scene or not is the reading period. To confirm if the tag is still the scene, the system uses the time of the last read tag, sum to the reading interval and to the reading period. The values are compare to the actual read time.

The equation above exemplifies the logic:

$$ActualTag(time) \leq LastTag(ms) + ReadingPeriod(ms) + ReadingTime(ms) \quad (5.1)$$

Besides all the parameters can be set to the 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, the error rate variable can be set and used in the permanence equation:

$$ActualTag(time) \leq LastTag(time) + ReadingPeriod(time) + ReadingTime(ms) + Errorrate(ms) \quad (5.2)$$

With the error rate in the equation, the operator certifies that, even with a delay in the reading step the permanence of an object will be more precise.

The value of error rate will vary and depend on the purpose of the timeline, it can also be used for other purposes than the delay problem. If the user want to keep the object in scene for 2 reading periods, for example, the error rate can be set to a value equal to Reading Period. Then, the Reading Period will be doubled.

After build the object timeline, the system only have 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.

In advanced studies and analysis, the back end has the registry of every read tag associated to its power in the moment of the read.

5.2 THE LOW-COST PROTOTYPE

Based on the functional blocks modularity, the system is also capable to be executed in a low-cost and portable system. The proposed prototype in this section is based on the Raspberry Pi device and an extension of the device implemented in (MORENO; MACHADO, 2013). The Raspberry Pi system is a low cost platform with educational purpose created by the Raspberry Pi foundation.

The Raspberry Pi is is a credit card size computer (85.60mm x 53.98mm x 17mm) equipped with two USB 2.0 ports, Ethernet interface, stereo audio, HDMI video interface, 512MB RAM memory and an 700MHz Arm processor, as shown in Figure 5.5. The indicated price for the product in the manufacturer site is US\$35,00. For this project we used one Raspberry Pi model B computer with one 4GB SD Card, one usb wireless card attached to the USB port and a serial to USB cable attached to the other USB port. The Raspberry Pi system is using a Linux Debian ARM compilation with all the necessary software installed.

Basically the prototype have the same functionality of the Full-fledged one. The Raspberry pi system is compatible with all the technologies used in the complete prototype such Java, PHP and Apache server. To the connection with the reader it was used one of the USB ports with the same USB to RS232 cable. The Raspberry Pi has a low processing hardware, in this context, the execution pattern module was suppressed to this prototype.

In this context, the low-cost prototype can not perform the distance inference pretended in the Full-fledged prototype. As a credit card size computer, this system can

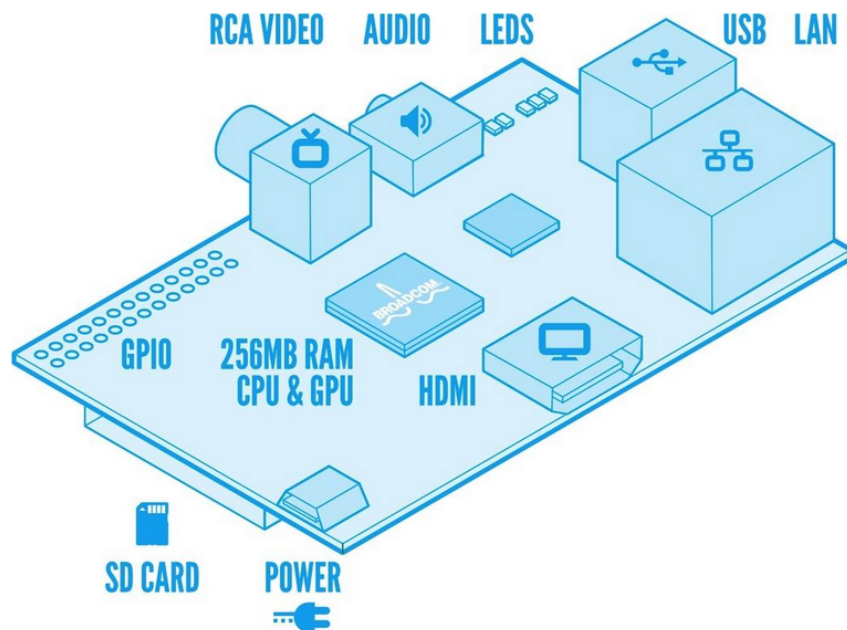


Figure 5.5: Raspberry Pi Architecture

also be a portable device to be attached to the RFID reader. The device can work as an wireless router device, in this way, in a recording studio the end user can connect to the device and control the front end.

5.3 OBJECT TIMELINE AND FRONT END

The object timeline is completely associated with the front end, which is the responsible for showing the end user the object timeline.

The front end is a web based solution, i.e., the end user 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 be easy to use, to handle large amounts of dynamic data, and to enable manipulation of and interaction with the data⁵.

As an example of the end-user visualization, Figures 5.6, 5.7, 5.8 and 5.9 show how the object timeline is operated in the front end. The example above represents a timeline for a scene with 3 objects. The scene duration is 8 seconds (from 1 to 9). Figure 5.6 shows the object timeline for the above situation:

- Object 1: Appear in scene in the fourth second and leave after three and a half

⁵www.visjs.org

second;

- Object 2: Appear in scene in the third second until the fifth one;
- Object 3: Is in scene from the start moment until the end. This object did not go out of the scene.

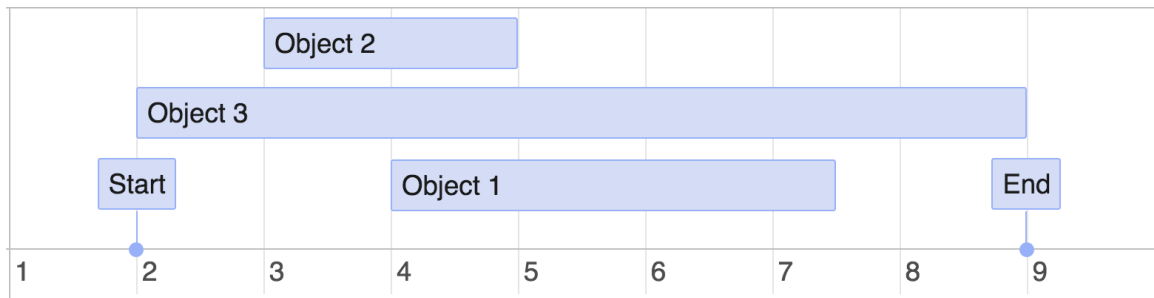


Figure 5.6: Object timeline example for Objects 1, 2 and 3.

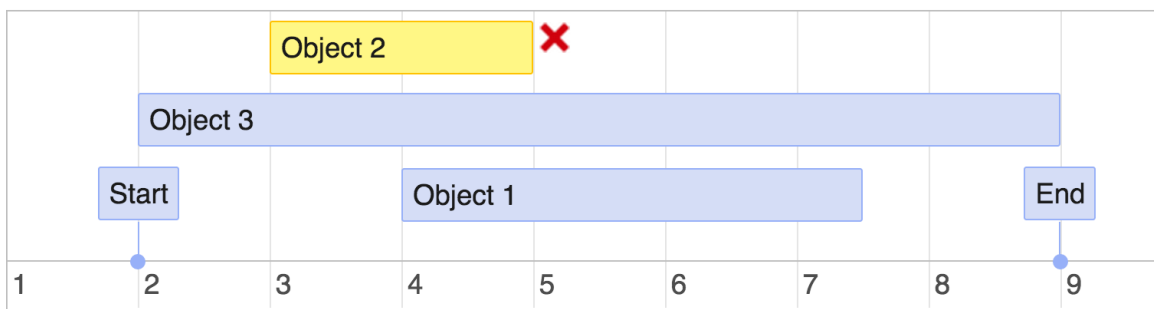


Figure 5.7: Object timeline - remove selection property

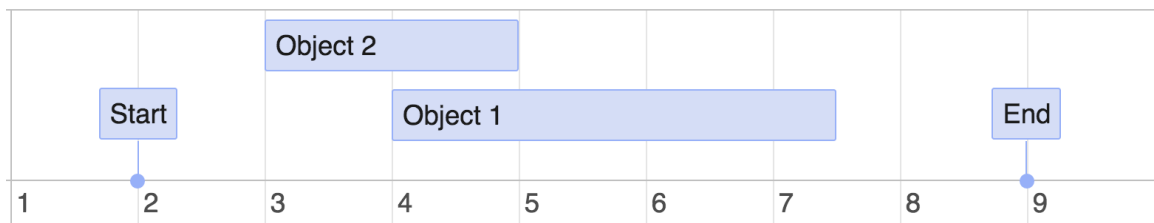


Figure 5.8: Object timeline: object 3 removal

Figure 5.7 illustrates the possibility to remove an object from the timeline with a simple method. The user just have to select the item and remove it in the "X". Figure 5.8 represents the object timeline after the removal of the Object 3.

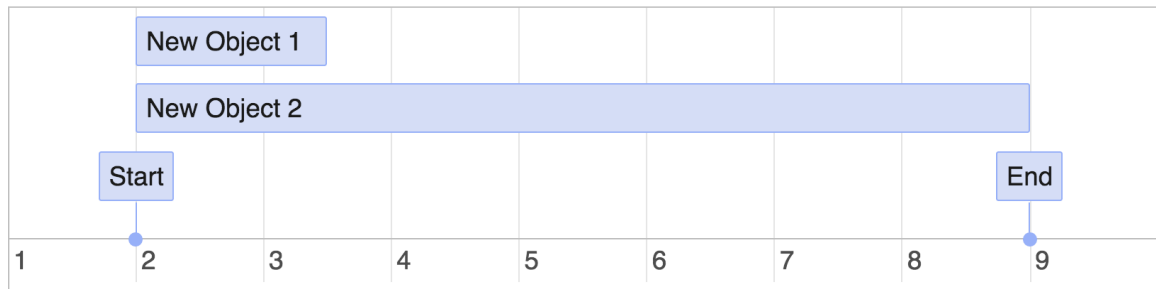


Figure 5.9: Object timeline - objects 1 and 2 edited

Other operations in front end can also be done. Figure 5.9 represents the final timeline, a simulation of a timeline edited by the end user. Object 3 was removed from the timeline, the name of objects 1 and 2 were changed and also their duration, start and end time. Object 1 had the permanence changed to one and a half second (from 1 to 2.5) and object 2 had the permanence changed to the entire scene (from 1 to 9).

Figure 5.10: Back end XML file (left column) and front end XML file (right column).

All the communication between the back end (Java) and the front end (PHP) is done by a WebSocket. WebSocket is a protocol that provides full-duplex communication channels over a single TCP connection (FETTE; MELNIKOV, 2011).

After the Back-end receives the identified tags and process the object timeline based on this, it send to the front end strings through the WebSocket. All the exchange messages between Back-end to front end are done in strings and operates in a way similar to web chats. The strings sent by the Back-end are strings compatible with the back end API, i.e., strings that update the objects in the timeline or operate the timeline.

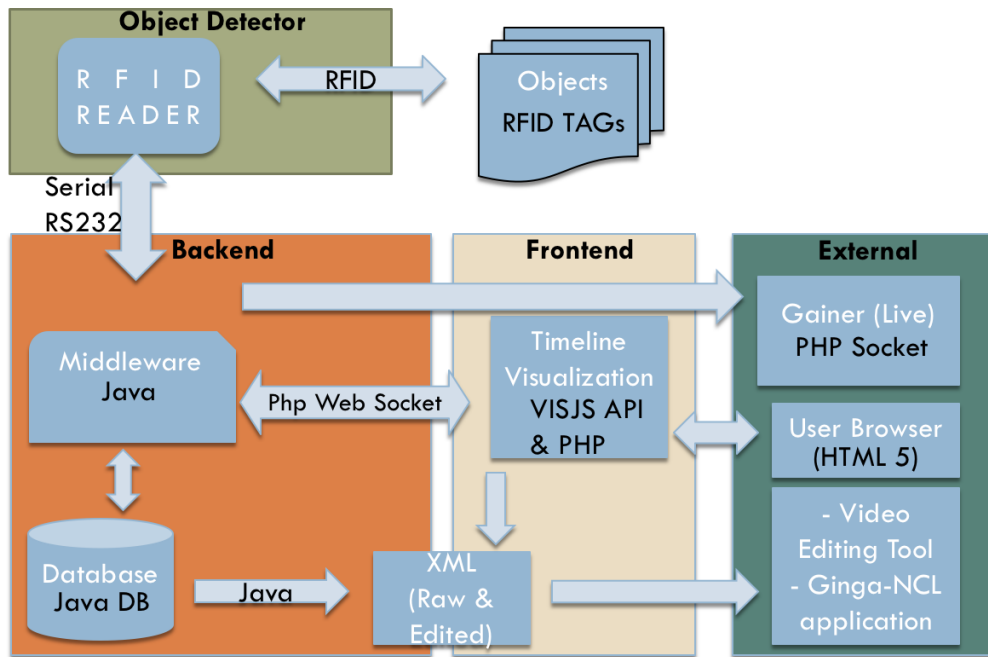


Figure 5.11: Implemented technologies used to RFID-based AODVP

The support for the alignment in the timeline is done by the register of the Start and End markers. Additionally the Start marker can bring informations of the job such work name, scene and take.

The object timeline output for the above example is shown in Figure 5.10. The image has both XML output file, the left column represents the raw object timeline, i.e., the original timeline represented in Figure 5.6. Moreover, the right column represent the output for the end timeline (5.9), after the edition by the end-user. It is important to notice that the edited timeline does not change the object object ID, even if the user changes the content.

In this chapter we presented the implementations used to create the prototype of the proposed architecture. Figure 5.11 summarize the technologies used based on the functional blocks and main architecture proposed on Chapter 4.

6 EXPERIMENTAL RESULTS

This chapter will describe the real tests executed to validate the proposed method for object detection, the object timeline.

To perform the object detection, a long range RFID reader was acquired. The model used is a UHF mid-range integrated reader (DLC-6820Z).

The reader is described as an RFID reader for tolling systems. According to its specification, in single tag read mode it can read tags in a speed up to 120 km/h. The recommended reading distance is from 5 to 10 meters. The frequency range is from 860 Mhz to 960 Mhz. The reader costs \$250.00. To connect with the reader, a USB to RS232 cable was used.

The reader did not come with any description of its working method and communication patterns, the first problem was to understand the reader functionalities and setup parameters, so, the first task was to map all the useful functionalities of the product that apply in this work. After that, it was necessary to sniff the RS232 serial port of the reader while it was in use. Then it was possible to create the pattern of the exchanged code between the software and the hardware. All the exchanged messages between the manufacturer software and the hardware are done with hexadecimal codes.

The patterns were mapped while using the reader. For each changed setup parameter it was possible to locate where the hexadecimal code was changed. Basically there are three kind of messages:

- Get messages: Requisitions from the middleware to the reader to get the value of the configured parameters on it;
- Set messages: Change the configured parameters of the reader;
- Response messages: For each Get/Set message the system send an error or success message.

Based on the mapped messages and the reader behavior it was possible to develop the system back end to control the reader and set the parameters as necessary.

The back end system was developed in Java. For the experiments, it was used a Virtual Machine with Windows XP installed. The virtual system runs over the Oracle

Virtual Box virtualization system. To the virtual machine 2 cores of an Intel i5 mobile processor were allocated. It was also set with 16mb of display memory, a 12GB hard disk and 2GB of RAM memory. The virtual machine was also set with an Apache server (version 2.2.21)¹ with PHP (version 5.3.10)².

The experimental results were made in three different environments. The first experiment was only built to perform the object detection, not considering the object timeline generation. The test is important to evaluate the reader as an object detector.

After that a full experiment was done, since the object detection to the object timeline and the XML file export.

Finally an example targeting an DTV application is build with NCL.

6.1 OBJECT DETECTION

The first step to create the object timeline is to detect the object in scene. The object detection is regardless to the object timeline creation. The stream to live broadcasters, for example, is only performed based on the object detection.

The object detection is also performed in the other examples of this Chapter. This section shows the object detection isolated from the object timeline to demonstrate the object detection performance.

To run the test, the reader was fixed in the ground level. All tags were placed in a range up to 1 meter from the reader and in a straight line from the reader. To avoid interference, none of the tags were placed linear to each other. The Figures 6.1, 6.2 and 6.3 exemplify the exact position of the tags in the three performed tests, respectively with three, four and five tags. In those tests, the tags were not placed into objects, the detection was made directly with the tags.

The results of the performed tests in these three environments are respectively represented in the the Table 6.1, Table 6.2 and Table 6.3. The detection have approximately 30 seconds, the reading interval was set to 6 seconds (6000 ms) and the reading period to 0.5 seconds (500ms). The first column is the time that the object was read and the second one the object itself. Each reading round is separated in the table by the line. In the first table, for example, the first reading round is from the second 2 to second 3.

¹<http://www.apache.org/>

²<https://www.php.net>

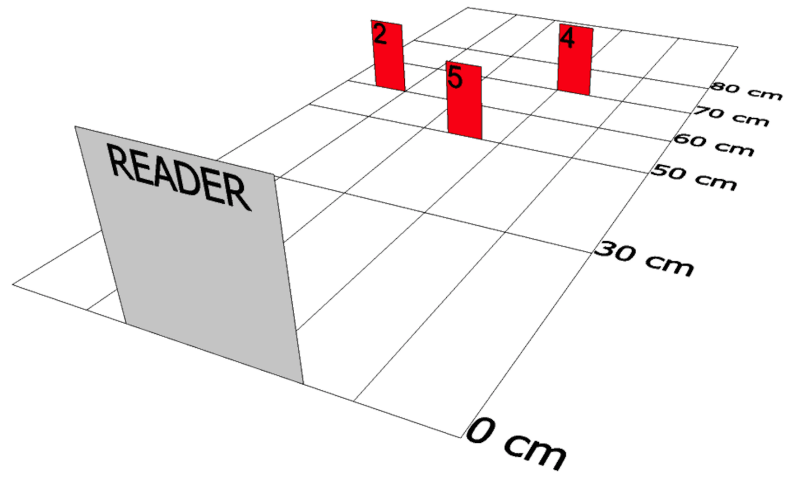


Figure 6.1: Object detection setting with 3 tags.

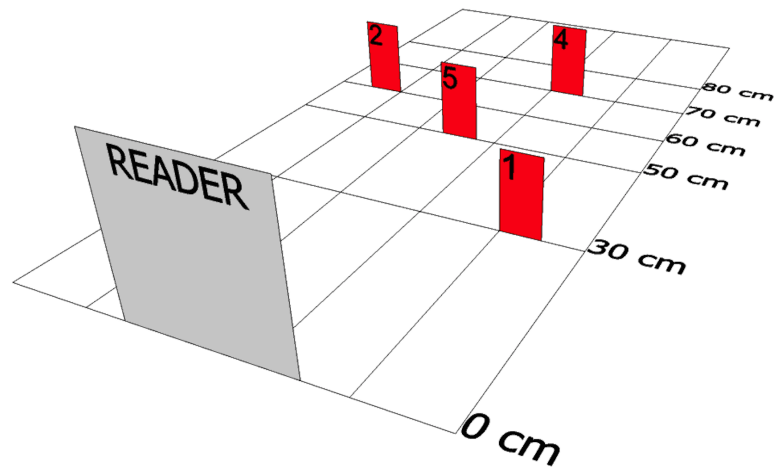


Figure 6.2: Object detection setting with 4 tags.

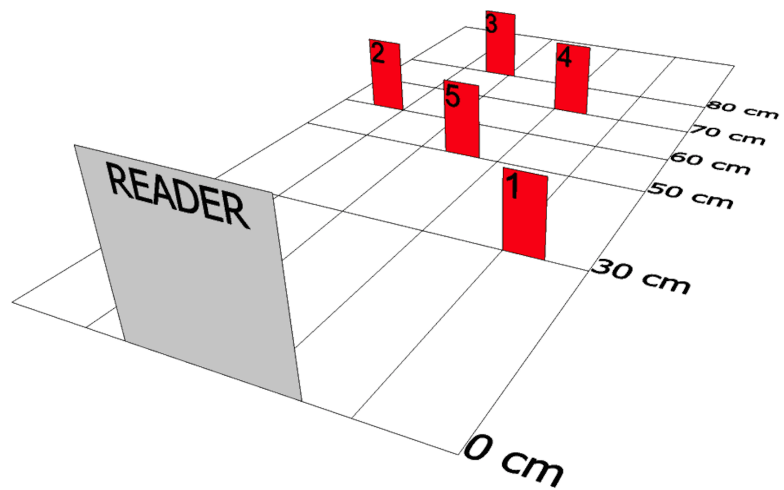


Figure 6.3: Object detection setting with 5 tags.

The next round is from the second 8 to 9. By that, the table is separated in 6 groups of reading, each group represents the reading interval of approximately 6 seconds.

Table 6.1: Object detection: 3 objects in 6 reading periods

Detect Time	Object	Detect Time	Object	Detect Time	Object
0:02.894	Object 4	0:02.694	Object 5	0:02.914	Object 2
0:02.934	Object 5	0:02.754	Object 2	0:02.954	Object 5
0:03.084	Object 2	0:02.884	Object 4	0:03.084	Object 4
0:08.813	Object 5	0:08.582	Object 2	0:08.832	Object 5
0:08.863	Object 2	0:08.642	Object 5	0:08.909	Object 2
0:08.943	Object 4	0:08.873	Object 4	0:09.427	Object 4
0:14.851	Object 5	0:14.661	Object 5	0:15.464	Object 5
0:14.891	Object 2	0:14.721	Object 2	0:15.530	Object 2
0:15.062	Object 4	0:14.891	Object 4	0:15.728	Object 4
0:20.940	Object 5	0:20.700	Object 2	0:21.909	Object 5
0:21.010	Object 4	0:20.750	Object 5	0:22.058	Object 2
0:21.110	Object 2	0:20.820	Object 4	0:22.216	Object 4
0:26.949	Object 5	0:26.688	Object 5	0:27.907	Object 5
0:27.079	Object 4	0:26.748	Object 2	0:27.947	Object 2
0:27.139	Object 2	0:27.209	Object 4	0:28.086	Object 4
0:33.058	Object 4	0:32.777	Object 5	0:33.426	Object 4
0:33.098	Object 2	0:32.847	Object 2	0:33.479	Object 5
0:33.228	Object 5	0:32.907	Object 4	0:33.524	Object 2

When the reader was used to read more than five simultaneous tags, the system rarely detect all of them. When using more than that, the system discards the extra ones. It is important to notice that the extra tag, is randomly not recognized, i.e., the system read five but it is not possible to determinate which tags will be identified in the group of five.

Thus, for a number of tags igual or less than five, all the tags were read in all reading rounds. Thus the reader accuracy was satisfactory to perform the future tests.

6.2 FULL OBJECT DETECTION

To perform the main experiment, a real use situation was created. The simulation was done with the reader positioned 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 others objects were placed between the reader and

Table 6.2: Object detection: 4 objects in 6 reading periods

Detect Time	Object	Detect Time	Object	Detect Time	Object
0:02.814	Object 5	0:02.814	Object 5	0:02.854	Object 2
0:02.864	Object 1	0:02.864	Object 1	0:02.904	Object 5
0:02.934	Object 2	0:02.934	Object 2	0:03.064	Object 1
0:03.085	Object 4	0:03.085	Object 4	0:03.214	Object 4
0:08.693	Object 1	0:08.693	Object 1	0:08.762	Object 5
0:08.823	Object 2	0:08.823	Object 2	0:08.832	Object 1
0:08.893	Object 5	0:08.893	Object 5	0:08.963	Object 2
0:09.544	Object 4	0:09.544	Object 4	0:09.053	Object 4
0:14.852	Object 2	0:14.852	Object 2	0:14.861	Object 5
0:14.912	Object 1	0:14.912	Object 1	0:14.931	Object 1
0:14.942	Object 5	0:14.942	Object 5	0:15.091	Object 2
0:15.022	Object 4	0:15.022	Object 4	0:15.312	Object 4
0:20.840	Object 1	0:20.840	Object 1	0:20.980	Object 2
0:20.880	Object 5	0:20.880	Object 5	0:21.030	Object 5
0:21.040	Object 2	0:21.040	Object 2	0:21.090	Object 1
0:21.181	Object 4	0:21.181	Object 4	0:23.383	Object 4
0:26.929	Object 5	0:26.929	Object 5	0:26.968	Object 1
0:26.969	Object 1	0:26.969	Object 1	0:27.119	Object 5
0:27.049	Object 2	0:27.049	Object 2	0:27.169	Object 2
0:27.109	Object 4	0:27.109	Object 4	0:27.800	Object 4
0:33.018	Object 1	0:33.018	Object 1	0:33.087	Object 5
0:33.078	Object 5	0:33.078	Object 5	0:33.137	Object 1
0:33.208	Object 2	0:33.208	Object 2	0:33.177	Object 2
0:33.348	Object 4	0:33.348	Object 4	0:33.358	Object 4

Table 6.3: Object detection: 5 objects in 6 reading periods

Detect Time	Object	Detect Time	Object	Detect Time	Object
0:02.894	Object 5	0:02.734	Object 1	0:02.804	Object 2
0:02.955	Object 1	0:02.794	Object 4	0:02.844	Object 5
0:03.045	Object 2	0:02.884	Object 5	0:02.884	Object 1
0:03.165	Object 4	0:02.994	Object 2	0:02.954	Object 4
0:03.285	Object 3	0:03.124	Object 3	0:03.034	Object 3
0:08.803	Object 4	0:08.632	Object 1	0:08.742	Object 4
0:08.903	Object 1	0:08.672	Object 5	0:08.783	Object 5
0:08.953	Object 5	0:08.782	Object 3	0:08.843	Object 1
0:09.083	Object 2	0:08.893	Object 4	0:08.953	Object 2
0:09.484	Object 3	0:08.963	Object 2	0:09.173	Object 3
0:14.892	Object 1	0:14.751	Object 1	0:14.781	Object 1
0:14.922	Object 5	0:14.841	Object 5	0:14.851	Object 2
0:15.022	Object 4	0:14.881	Object 3	0:14.951	Object 5
0:15.092	Object 2	0:15.001	Object 4	0:15.011	Object 4
0:15.513	Object 3	0:15.112	Object 2	0:15.182	Object 3
0:21.021	Object 3	0:20.820	Object 3	0:20.890	Object 2
0:21.081	Object 1	0:20.870	Object 1	0:20.930	Object 5
0:21.111	Object 5	0:20.990	Object 4	0:20.970	Object 1
0:21.201	Object 4	0:21.050	Object 2	0:21.100	Object 3
0:21.311	Object 2	0:21.090	Object 5	0:21.210	Object 4
0:27.019	Object 2	0:26.888	Object 1	0:27.019	Object 2
0:27.089	Object 1	0:26.969	Object 5	0:27.089	Object 1
0:27.139	Object 5	0:27.049	Object 3	0:27.209	Object 5
0:27.229	Object 4	0:27.119	Object 4	0:27.279	Object 3
0:27.290	Object 3	0:27.199	Object 2	0:27.369	Object 4
0:33.148	Object 5	0:32.907	Object 5	0:33.077	Object 4
0:33.198	Object 1	0:32.957	Object 1	0:33.188	Object 5
0:33.278	Object 2	0:33.047	Object 3	0:33.258	Object 3
0:33.368	Object 4	0:33.117	Object 2	0:33.338	Object 1
0:33.478	Object 3	0:33.178	Object 4	0:33.418	Object 2

the monitor (MACHADO, 2015).

The idea was to perform a test that encompassed all use scenarios. The control of the scene is done by the 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 been in scene, the CD Cover is not registered at the Java DB, i.e., it should be read but not registered in the timeline. Other objects will enter and leave the scene. Table 6.4 describe all the objects (name, tag number and tag functionality) used to perform the experiment.

Table 6.4: Objects used to real simulation.

Object	ID (first 4 digits)	Functionality	Registered to 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 resumed below:

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

Figure 6.4 illustrate the real scenario of the test.

For the created scenario, the 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 listened in the object timeline. As described in Table 6.4, the object



Figure 6.4: Real environment.

CD is not registered in the database, in this case, it should not be in timeline, even been detected by the reader.

Tables 6.5 and 6.6 resumes the object detection for the registered and non-registered objects.

Table 6.5: Registered object detection

Detect Time	ID	Description
0:02.038	99992064790602721860549e	LCD Monitor
0:07.508	99992064790602721860549e	LCD Monitor
0:11.484	7777207561120046213037a5	Clapperboard
0:12.981	99992064790602721860549e	LCD Monitor
0:13.021	7777207561120046213037a5	Clapperboard
0:18.519	7777207561120046213037a5	Clapperboard
0:18.965	99992064790602721860549e	LCD Monitor
0:24.008	99992064790602721860549e	LCD Monitor
0:29.396	33332075611400951080a86c	WALLET
0:30.388	99992064790602721860549e	LCD Monitor
0:30.456	33332075611400951080a86c	WALLET
0:36.126	33332075611400951080a86c	WALLET
0:36.666	99992064790602721860549e	LCD Monitor
0:42.572	33332075611400951080a86c	WALLET
0:42.680	99992064790602721860549e	LCD Monitor
0:46.219	88882064790602690980b36a	BOOK
0:48.723	33332075611400951080a86c	WALLET
0:48.783	88882064790602690980b36a	BOOK
0:49.284	99992064790602721860549e	LCD Monitor
0:55.302	33332075611400951080a86c	WALLET
0:55.833	88882064790602690980b36a	BOOK
0:55.903	99992064790602721860549e	LCD Monitor
1:01.361	88882064790602690980b36a	BOOK
1:01.892	99992064790602721860549e	LCD Monitor
1:07.880	88882064790602690980b36a	BOOK
1:07.951	99992064790602721860549e	LCD Monitor
1:12.828	555520647906027318904c89	PS3 Controller

For the described scene and objects detection, Figure 6.5 shows the end-user final visualization to the object timeline.

As expected, the LCD Monitor object was identified in the first moment of the capture but its presence in timeline was determinate by the Start-Tag (Clapperboard). As Table 6.5 shows, the system identified two occurrences of the LCD Monitor before the

Table 6.6: Unregistered objects detection

Detect Time	ID	Description	Real interval
0:02.093	222220756114002515207bb4	CD	-
0:07.003	222220756114002515207bb4	CD	4.910
0:13.517	222220756114002515207bb4	CD	6.514
0:18.484	222220756114002515207bb4	CD	4.967
0:23.949	222220756114002515207bb4	CD	5.465
0:29.867	222220756114002515207bb4	CD	5.918
0:36.083	222220756114002515207bb4	CD	6.216
0:42.702	222220756114002515207bb4	CD	6.619
0:49.344	222220756114002515207bb4	CD	6.642
0:55.392	222220756114002515207bb4	CD	6.048
1:01.401	222220756114002515207bb4	CD	6.009
1:07.380	222220756114002515207bb4	CD	5.979

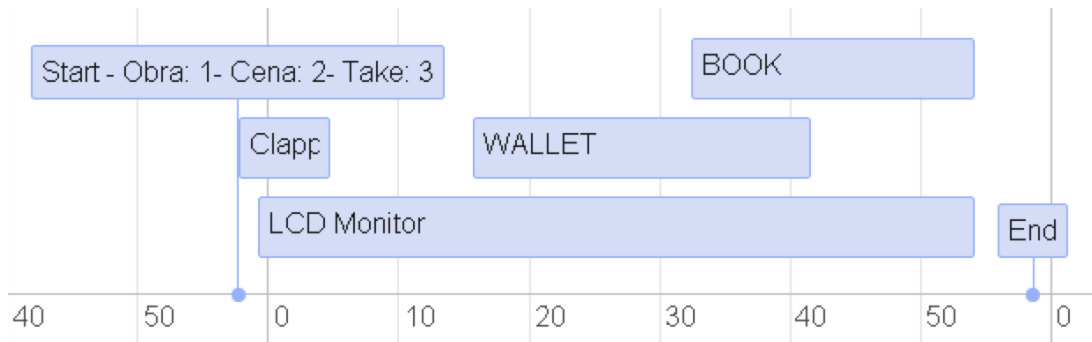


Figure 6.5: Final timeline for the real usage experiment.

Clapperboard, but the object was just sent to the object timeline after the Clapperboard object, as exemplified in Figure 6.5. The tag was also present in all the captured timeline. 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 appear in scene and does not leave until the end.

According to Table 6.6, in the last column, the reading interval between each object identification was close to 6 seconds. As described before, the reading interval in the experiment was set to 6 seconds and the reading time to 0.5 seconds. Applying those parameters to the Equation 5.1, the last read of a tag compared to the actual one could not be greater than 6500 ms, in this case, the back end stop the object in the last read and register another entry to the object.

In the experiment, the object CD was in scene at all the time. Although not being registered in the database, it was read in every read period of the reader. As illustrated

```

exp3_front.xml
1 <objects>
2 <object>
3 <id>57</id>
4 <content>Start - Obra: 1- Cena: 2- Take: 3</content>
5 <start>2015-08-18T00:35:57.720Z</start>
6 </object>
7 <object>
8 <id>Clapperboard</id>
9 <content>Clapperboard</content>
10 <start>2015-08-18T00:35:57.740Z</start>
11 <end>2015-08-18T00:35:57.740</end>
12 <power>0</power>
13 </object>
14 <object>
15 <id>LCD Monitor</id>
16 <content>LCD Monitor</content>
17 <start>2015-08-18T00:35:59.227Z</start>
18 <end>2015-08-18T00:36:54.207Z</end>
19 <power>0</power>
20 </object>
21 <object>
22 <id>WALLET</id>
23 <content>WALLET</content>
24 <start>2015-08-18T00:36:15.652Z</start>
25 <end>2015-08-18T00:36:41.548Z</end>
26 <power>0</power>
27 </object>
28 <object>
29 <id>BOOK</id>
30 <content>BOOK</content>
31 <start>2015-08-18T00:36:32.475Z</start>
32 <end>2015-08-18T00:36:54.116Z</end>
33 <power>0</power>
34 </object>
35 <object>
36 <id>1439869018707</id>
37 <content>End</content>
38 <start>2015-08-18T00:36:58.653Z</start>
39 </object>
40 </objects>

exp3_back.xml
1 <objects>
2 <object>
3 <id>57</id>
4 <content>Start - Obra: 1- Cena: 2- Take: 3</content>
5 <start>2015-08-18T00:35:57.720</start>
6 </object>
7 <object>
8 <id>Clapperboard</id>
9 <content>Clapperboard</content>
10 <start>2015-08-18T00:35:57.740</start>
11 <end>2015-08-18T00:35:57.740</end>
12 <power>0</power>
13 </object>
14 <object>
15 <id>LCD Monitor</id>
16 <content>LCD Monitor</content>
17 <start>2015-08-18T00:35:59.227</start>
18 <end>2015-08-18T00:36:54.207</end>
19 <power>0</power>
20 </object>
21 <object>
22 <id>WALLET</id>
23 <content>WALLET</content>
24 <start>2015-08-18T00:36:15.652</start>
25 <end>2015-08-18T00:36:41.548</end>
26 <power>0</power>
27 </object>
28 <object>
29 <id>BOOK</id>
30 <content>BOOK</content>
31 <start>2015-08-18T00:36:32.475</start>
32 <end>2015-08-18T00:36:54.116</end>
33 <power>0</power>
34 </object>
35 <object>
36 <id>1439869018707</id>
37 <content>End</content>
38 <start>2015-08-18T00:36:58.653</start>
39 </object>
40 </objects>

```

Figure 6.6: XML output file from the back end (left side) and front end (right side).

in Table 6.6, there are three reading intervals higher than 6500 ms (6514, 6618 and 6642), those values are highlighted in bold.

Based on Equation 5.1 and if the object CD was registered, the timeline would not consider the permanence of the object from the beginning to the end. In this case the timeline would register the object with three appearances in scene, not corresponding to the real experiment.

Been the RFID a wave propagation technology, highly susceptible to interference and, as experiments have shown that in many cases the Equation 5.1 is not valid, the error rate should be use. In the analyzed case, with an error rate of 250 ms, i.e., less than 4% of the total time, the problem could be solved. Applying in Equation 5.2, the time to create a new object in scene and do not continue the existent one increase to 6750 ms. This value is greater than all the values in the Table 6.6.

Finally, the XML output files from the back end and the front end are shown in Figure 6.6, in this case, both XML archives have the same content. For the experiment, the user did not interfered in the timeline by the use of the front end.

The real experiment and system performance occurs just as expected. The results have shown that, as a concept proof, the system worked as predicted with no errors. It is also clear the importance of using the error rate to improve the results in real situations.

6.3 ADVANCED FEATURES

As suggested in this work, the reader was also set to infer distance by the implemented power variation function.

To avoid radio interference, those tests were made in open field. The reader was positioned with a 90 degree angle to the ground. It was also pointed directly to the tags, which were in a distance from 10cm to the floor, i.e., the distance of the tags to the the floor was the same distance from the floor to middle of the reader.

The execution pattern was set to vary the power in a step of 5. The interval was set to 10 seconds. For distance infer, any method of variation can be used, we adopt the CZC method. The reading time was set to 0.01 second and period interval to 15 seconds. All the tests were made only with one tag.

The following parameters were used to validate the collected data: The reader should read the tag in 10 consecutive reads. We perform 50 test rounds.

Unfortunately, the description of the RFID reader does not specify the real read distance, how does the power variation works and the position of the reader antenna.

Table 6.7 resume the collected data. Although the specification of the reader determines that the power varies from 1 to 150, the reader could not read any tag with the power set to any value less than 80. It also did not read any tag in a distance superior to 2 meters. Accordingly to the reader seller, the equipment was capable to read passive tags up to 5 meters. As explained, the tags and the reader were positioned in a straight line. This highlights the problem of not having a correct documentation of the hardware.

Furthermore, when the power was set to values from 130 to 150, the reader could exactly read the tags in a distance superior to 2 meters. Tags in a distance superior than 170cm can be in any power above 130.

The distance inference tests were made in a very specific scenery to avoid inaccurate results. Another problem is that the code developed to perform the power variation in the RFID reader can not handle with the slowness of the reader hardware. When trying to perform the power variation and the tag read, it was necessary to set the reading

Table 6.7: Reader distance from power variation

Power	Distance(cm)
80	0
85	5
90	55
95	75
100	90
110	150
120	170
130	200
140	200
150	200

parameters and the power variation to greater values, avoiding any kind of system lock. When the reader is set to read tags in time lower than 10 seconds, the RFID reader could not perform precisely the power variation, this because there is no way to perform those operations in different threads.

Despite these limitations of the reader and even forcing one specific reading scenario, the results have shown that the RFID technology can also be used to perform distance inference.

6.4 OBJECT TIMELINE DELIVERING WITH NCL

In the field of digital and interactive TV, 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 (SOARES et al., 2010).

According to Soares et al. (2010), NCL documents do not contain multimedia elements such as audio or video content; rather they function as a "glue" language that specifies how multimedia components are related. In particular, NCL documents specify how these components are synchronized relative to each other and how the components are composed together into a unified document. Among its main facilities, it treats hypermedia relations as first-class entities through the definition of hypermedia connectors, and it can specify arbitrary semantics for a hypermedia composition using the concept of composite templates.

³www.ncl.org.br

NCL was initially designed for the Web environment, but a major application of NCL is use as the declarative language of the Japanese-Brazilian ISDB-Tb (International Standard for Digital Broadcasting) terrestrial DTV digital television middleware (named Ginga⁴). It is also the first standardized technology of the ITU-T multimedia application framework series of specifications for IPTV (internet protocol television) services. In both cases it is used to develop interactive applications to digital television (FILHO et al., 2007).

In order to exemplify the use of an external agent for the object timeline, we develop an NCL application based on the main experiment. The XML generated in the object detection (Figure 6.6) and the video recorded (MACHADO, 2015), were used in this experiment in a simulation of real use in DTV.

A T-Commerce (television commerce) template was used in this example. This template allows the user to buy products as they appear in the TV. The viewer can use the color buttons in the DTV controller to interact with the NCL application.

The items only appear in the TV for the viewer as they are detected in scene. Figures 6.7, 6.8, 6.9, 6.10 and 6.11 exemplify all the object detection and the interaction of the viewer in the DTV application.

Figure 6.7 represents the start of the scene, at this point only the monitor was detected in scene. The DTV application offers the product to be brought by using the red button in DTV controller. Figure 6.8 represents the moment that the wallet appears in scene, in the same way the user can use the controller to buy the product.

The experiment simulates a wallet purchase by the user (Figure 6.10). The product is added to cart and the buyer can checkout, as Figure 6.11 represents. At this point the wallet is out of the scene, in this way, the product is not offered again.

⁴www.ginga.org.br



Figure 6.7: NCL example, LED monitor offer



Figure 6.8: NCL example, LED monitor and wallet offer.



Figure 6.9: NCL example, LED monitor, book and wallet offer.



Figure 6.10: NCL example, LED monitor, book and wallet offer. Wallet added to cart.



Figure 6.11: NCL example, LED monitor and book offer. Checkout enable.

To produce this experiment the object timeline XML was mapped into the NCL editor with markers. Based on these markers, the NCL trigger the exhibition or the removal of an item from the list of products available to purchase with the use of *links*. Figure 6.12 illustrate the NCL media component, the description of the elements and time of their appearance in scene.

```
<media id="videoPrincipal" src="\ExperimentoRFID.mp4" descriptor="descVideo">
  <area id="ancLCDMONITOR" begin="1.5s" end="56.5s"/>
  <area id="ancWALLET" begin="18s" end="44s"/>
  <area id="ancBOOK" begin="35s" end="56.5s"/>
</media>
```

Figure 6.12: Anchors markers from the object timeline XML.

The *link* used to show a product offer is illustrated in Figure 6.13. The removal of the same item is illustrated in Figure 6.14.

```
<link xconnector="OnBeginStartNSetNStopN">
  <bind role="onBegin" component="videoPrincipal" interface="ancWALLET"/>
  <bind role="start" component="ctxWALLET" interface="portWALLET"/>
</link>
```

Figure 6.13: Link exhibition for the wallet object.

```
<link xconnector="onEndStopNSetN">
  <bind role="onEnd" component="videoPrincipal" interface="ancWALLET"/>
  <bind role="stop" component="ctxWALLET"/>
</link>
```

Figure 6.14: Link removal for the wallet object.

This section presented a DTV application based in NCL that offer products to the user based on their appearance in the scene, i.e., their detection. The example shows that the use of automated content can easily be done based on the object timeline XML file.

7 CONCLUSION

In this work, we have addressed an automated object detector to be used during video production, specially in Digital TV context. Our intend was to provide a robust and trustable platform to the broadcaster, targeting the so-called Automated Object Detection in Video Production (AODVP) and delivering this information to video editing software and iDTV authoring tools, in an easy way to be handled.

First we proposed a technology-free architecture of an AODVP system. The functional blocks were described and their functionalities.

Based on the proposed architecture we develop our apparatus. First a RFID object detector was used to do the object identification in scene. Once the object detector was selected, we develop the functional blocks system 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 object 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 receivers as a XML file.

We also performed the validation of the XML file, developing an example of an interactive Digital TV application with NCL language. Based on a T-Commerce template, the application import the XML file and, in the moment that the object appearance in scene, the product is offered to the televiewer. Using the iDTV controller and its color buttons (red, blue, yellow and green), the user can buy the offered product.

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 exceed five. As the proposed example of an NCL application uses 4 objects, this limitation did not detracted the results.

As an advanced feature a distance inference test was executed with the RFID reader. The results have shown that the RFID technology is also promising in this field. However, the experiments were done in a controlled environment that do not represent real

situations.

Advances in the study of the metadata, specially in objects context can also be done. As the system provide the information of the objects in scene, these objects can be automatic correlated. Hereupon, the system can automatic provide the limit age for the televiewer to access the content, for example.

In the field of the Object Detector, we propose advanced studies in RFID technology, specially in its propagation signal, thus better results can be achieved, and also a more accurate distance inference. It is also interesting to develop experiments in advanced RFID readers, with the possibility to control the antennas, a better control of the power (gain) and the parameters on it. It is also important to improve the use of this technology, specially when dealing with interference. . Other system can also be used to perform the object detection, whereas it must work as a real time technology.

Another interest improve in the system is to use more than one RFID reader at the same time. To a better improve of the distance inference, a distance matrix can be created to precise the distance of the tag.

It is also important to notice that the RFID reader used in this work is specific built to work in toll system. In this way, the hardware was not built to detect too many tags in a small reading range. The reader work in the so-called reading upon ID difference. In this mode, the reader is always reading, it is not necessary to set up the reading duration and interval time. When always reading, it will detect any tag in the range, but it does not read one tag two or more times. This mode is set to work in tolls system where it can not detect one car two times consecutively.

With more advanced control of the RFID reader it will be possible to integrate the object timeline capture to the video camera control. This advance can provide, for example, the focus of the object detection. If the camera focus an object in the back of the studio or close to the camera, the reader can change its configuration to only capture the object in camera range.

This work proposes a new technology to be used in video recording studios for the automation of interactive content production. We proposed the general framework and also an effective example of an apparatus that can be used from the detection of the object to the creation of the interactive content, all automated.

REFERENCES

- ABRAHAM, S. M.; BIJLANI, V. A.; THOMAS, M. **Method, medium, and system for automatically embedding information concerning items appearing in video using RFID tags**, abr. 26 2011. US Patent 7,933,809.
- ALVES, L. G. P.; KULESZA, R.; SILVA, F. d.; JUCÁ, P.; BRESSAN, G. Análise comparativa de metadados em tv digital. In: **II Workshop de TV Digital do Simpósio Brasileiro de Redes de Computadores**, 2006.
- BERRY, M.; YANG, C. **Timeline alignment for closed-caption text using speech recognition transcripts**, 2012. US Patent 8,281,231. Disponível em: <<http://www.google.com.ar/patents/US8281231>>.
- CAUBERGHE, V.; PELSMACKER, P. D. Opportunities and thresholds for advertising on interactive digital tv: A view from advertising professionals. **Journal of interactive advertising**, Taylor & Francis, v. 7, n. 1, p. 2–23, 2006.
- CHOW, H. K.; CHOY, K. L.; LEE, W.; LAU, K. Design of a rfid case-based resource management system for warehouse operations. **Expert systems with applications**, Elsevier, v. 30, n. 4, p. 561–576, 2006.
- CROCKFORD, D. Rfc 4627. **The application/json Media Type for JavaScript Object Notation**, 2006.
- DACONTA, M. C.; OBRST, L. J.; SMITH, K. T. **The Semantic Web: a guide to the future of XML, Web services, and knowledge management**, 2003.
- DEDES, G.; DEMPSTER, A. G. Indoor gps positioning. In: CITESEER. **Proceedings of the IEEE Semiannual Vehicular Technology Conference**, 2005.
- DURAND, G.; KAZAI, G.; LALMAS, M.; RAUSCHENBACH, U.; WOLF, P. A meta-data model supporting scalable interactive tv services. In: IEEE. **Multimedia Modelling Conference, 2005. MMM 2005. Proceedings of the 11th International**, 2005. p. 386–391.

- FASEL, I.; FORTENBERRY, B.; MOVELLAN, J. A generative framework for real time object detection and classification. **Computer Vision and Image Understanding**, Elsevier, v. 98, n. 1, p. 182–210, 2005.
- FETTE, I.; MELNIKOV, A. Rfc 6455: The websocket protocol. **IETF**, **December**, 2011.
- FILHO, G. L. d. S.; LEITE, L. E. C.; BATISTA, C. E. C. F. Ginga-j: The procedural middleware for the brazilian digital tv system. **Journal of the Brazilian Computer Society**, SciELO Brasil, v. 12, n. 4, p. 47–56, 2007.
- GAVRILA, D. M.; PHILOMIN, V. Real-time object detection for “smart” vehicles. In: **IEEE. Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on**, 1999. v. 1, p. 87–93.
- GLOBO, R. **Revirando o Baú: Robô localiza fitas nos arquivos da TV Globo**, 2009. Disponível em: <<http://globotv.globo.com/rede-globo/video-show/v/revirando-o-bau-robo-localiza-fitas-nos-arquivos-da-tv-globo/1082508/>>.
- GONZALEZ, H.; HAN, J.; LI, X.; KLABJAN, D. Warehousing and analyzing massive rfid data sets. In: **IEEE. Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on**, 2006. p. 83–83.
- GRABNER, H.; GRABNER, M.; BISCHOF, H. Real-time tracking via on-line boosting. In: **BMVC**, 2006. v. 1, n. 5, p. 6.
- JAIN, R.; KASTURI, R.; SCHUNCK, B. G. **Machine vision**, 1995.
- JUELS, A. One billion digital tv homes by year-end. **Selected Areas in Communications, IEEE Journal on**, v. 1, n. 2, 2014.
- KOPETZ, H. **Real-time systems: design principles for distributed embedded applications**, 2011.
- KUMAR, V.; FURUTA, R.; ALLEN, R. B. Metadata visualization for digital libraries: interactive timeline editing and review. In: **ACM. Proceedings of the third ACM conference on Digital libraries**, 1998. p. 126–133.
- LANDT, J. The history of rfid. **Potentials, IEEE, IEEE**, v. 24, n. 4, p. 8–11, 2005.

- LOWE, D. G. Distinctive image features from scale-invariant keypoints. **International journal of computer vision**, Springer, v. 60, n. 2, p. 91–110, 2004.
- LOWE, D. G. **Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image**, 2004. US Patent 6,711,293.
- LUGMAYR, A.; NIIRANEN, S.; KALLI, S. Digital interactive tv and metadata. **Tampere University of Technology, Finland**, Springer, 2004.
- MACHADO, T. **Real Object Detect**. 2015. Disponível em: <<https://www.youtube.com/watch?v=wewQ1UwRKTE>>.
- MARIANANTONI, A.; PARK, H.; FRIEDMAN, J.; HOLTGREWE, V.; BURKE, J.; SRIVASTAVA, M.; WAGMISTER, F.; MCDONALD, W.; BRUSH, J. Sensor networks for media production. In: ACM. **Proceedings of the 2nd international conference on Embedded networked sensor systems**, 2004. p. 325–325.
- MERTZ, C.; NAVARRO-SERMENT, L. E.; MACLACHLAN, R.; RYBSKI, P.; STEINFELD, A.; SUPPE, A.; URMSON, C.; VANDAPEL, N.; HEBERT, M.; THORPE, C. et al. Moving object detection with laser scanners. **Journal of Field Robotics**, Wiley Online Library, v. 30, n. 1, p. 17–43, 2013.
- MIKOLAJCZYK, K.; TUYTELAARS, T.; SCHMID, C.; ZISSERMAN, A.; MATAS, J.; SCHAFFALITZKY, F.; KADIR, T.; GOOL, L. V. A comparison of affine region detectors. **International journal of computer vision**, Springer, v. 65, n. 1-2, p. 43–72, 2005.
- MORENO, M. F.; MACHADO, T. Sistema de baixo custo para a detecção de objetos reais na produção do audiovisual. **Revista Lumina**, v. 7, n. 2, 2013. ISSN 1981-4070.
- MURRAY, s. Rfid security and privacy: A research survey. **www.digitaltvresearch.com**, IEEE, v. 24, n. 1, p. 340, 2006.
- OH, J.; KIM, G.; PARK, J.; HONG, I.; LEE, S.; YOO, H.-J. A 320mw 342gops real-time moving object recognition processor for hd 720p video streams. In: IEEE. **Solid-State Circuits Conference Digest of Technical Papers (ISSCC), 2012 IEEE International**, 2012. p. 220–222.

- ROBERTS, C. M. Radio frequency identification (rfid). **Computers & Security**, Elsevier, v. 25, n. 1, p. 18–26, 2006.
- SANTINI, A. G. Rfid: Conceitos, aplicabilidades e impactos. **Rio de Janeiro: Ciência Moderna**, 2008.
- SHOTTON, J.; SHARP, T.; KIPMAN, A.; FITZGIBBON, A.; FINOCCHIO, M.; BLAKE, A.; COOK, M.; MOORE, R. Real-time human pose recognition in parts from single depth images. **Communications of the ACM**, ACM, v. 56, n. 1, p. 116–124, 2013.
- SIGEL, K.; DEANGELIS, D.; CIHOLAS, M. **Camera with object recognition/data output**, abr. 8 2003. US Patent 6,545,705.
- SOARES, L. F.; MORENO, M. F.; NETO, C. D. S. S. Ginga-ncl: declarative middleware for multimedia iptv services. **Communications Magazine, IEEE**, IEEE, v. 48, n. 6, p. 74–81, 2010.
- SOARES, L. F. G.; FILHO, G. L. d. S. Interactive television in brazil: System software and the digital divide. In: **European Conference on Interactive TV (EuroITV)**, **Áustria**, 2007.
- STANDARDIZATION, I. O. for. **Coding of Moving Pictures and Audio**, 2013. JTC1, SC29, WG11.
- STEVENS, T. **Inside Volvo self-driving car: Improving driver safety without the driver**. 2015. Disponível em: <<http://www.cnet.com/news/a-ride-in-volvos-autonomous-car-how-the-next-step-in-driver-safety-requires-replacing-the-driver/>>.
- VIOLA, P.; JONES, M. Robust real-time object detection. **International Journal of Computer Vision**, v. 4, p. 51–52, 2001.
- WU, J.; CUI, Z.; SHENG, V. S.; ZHAO, P.; SU, D.; GONG, S. A comparative study of sift and its variants. **Measurement Science Review**, v. 13, n. 3, p. 122–131, 2013.
- YANG, P.; WU, W.; MONIRI, M.; CHIBELUSHI, C. C. Efficient object localization using sparsely distributed passive rfid tags. **Industrial Electronics, IEEE Transactions on**, IEEE, v. 60, n. 12, p. 5914–5924, 2013.

ZHANG, Z. Microsoft kinect sensor and its effect. **MultiMedia, IEEE, IEEE**, v. 19, n. 2, p. 4–10, 2012.