A Framework Based on Image Processing Techniques for Providing Synchronization in Smart TV

Cédrick Bamba Nsimba¹ and Alexandre Luis Magalhães Levada²

¹Department of Computer Science, Federal University of São Carlos, São Carlos, São Paulo, Brazil
²Department of Computer Science, Federal University of São Carlos, São Carlos, São Paulo, Brazil

Abstract—In smart TV, we can naturally observe a lack of connection between applications and the content of the tuned programming on TV set and TV broadcasters have fully control over the transmitted content. Based on the fact that computer vision and machine learning tools can provide that synchronized information by, respectively, processing the image frame of TV content and then classifying them, new opportunities will be opened up for developing a bunch of new interesting applications aiming to promote the user interaction level in TV. Moreover, with the extensive use of mobile devices and computers in today’s life, it will have new user interaction possibilities and a new business model will be emerged. In this paper, the smart TV framework is presented and evaluated. The objective of this study is to facilitate the developers to implement applications in this area without being concerned about low-level implementation details.

Keywords: TV channels monitoring, image classification, multimedia synchronization, detection of TV channel logos, smart TV

Type of the submission: Regular Research Paper

1 Introduction

Smart TV is changing the way that people watch television and providing to the user a possibility to interact with TV contents. However, unlike this type of interaction, most of the time, the audiovisual content of programs in the smart TV does not communicate with the native TV applications and other TV platform features. The only basic interaction is the possibility to switch between the things are displaying on the TV screen either program or TV applications.

Bachelet [1] conducted a research in five European and North American countries in May 2013 and found out that only less than the half of 6115 smart TV owners who were interviewed, connected their TVs to the Internet. According to his research, two elements are the reasons that why the smart TV has not yet become popular and been widely accepted by consumers as much as smartphones: (a) The lack of content and interesting applications: although the majority of smart TVs offer a wide range of content and applications, most of them are irrelevant and are not interesting to users; and (b) The Poor User Interface: a lack of rich user interface that can integrate TV applications with its audiovisual contents.

Schofield [12] mentioned, “If TVs are going to be truly smart they must do more than offer a wide variety of online video services. Instead they must add advanced functionality including voice control, motion control, advanced advertising, attractive user interfaces and two-way communications with other smart devices –so-called ‘second screens’— allowing these devices both to send video to the TV and know what is being watched. Manufacturers should focus less on adding more content and more on improving how users can interact with that content”.

This suggests that new synchronization mechanisms, also interaction with TV environment can help to improve the user experience. Information related to the channel being watched by the user can be retrieved automatically through a clear approach using TV channel logo detection, for example. Then, this information can be used to notify all connected devices, in the environment of smart TV, about the channel being watched by the user. Channel logos are visual objects (name, symbol or trademark) designed with the purpose of easy recognition and are important to assess the identification of a TV channel.

One of the finding mentions that innovative interactivity has not been promoted and used adequately in smart TV platforms. In addition, they show that the lack of facilities that allow the integration of Smart TV applications with their own audiovisual content is a major reason for its limited use. Consequently, the current proposed paper uses image processing to promote the development of smart TV applications synchronized with TV program contents.

2 Related Work

Several authors reported some results of building frameworks designed to support TV applications development. However just few of them focus on reuse in smart TV applications, specifically those ones synchronized with television programming.

Group Share-TV [6] proposes a framework called share-TV used for the development of converged applications centered on TV for GoogleTV and Ginga-J platforms. The share-TV allows the development of TV applications that include a generic mobile application. Ever since this generic application get downloaded from a given available TV IP,
installed and run on mobile device, the communication with the TV convergent application will be initiated automatically in order to register and receive shared objects. While the objects are being received, the device show them on the screen allowing interaction on them. Compared to the work reported in this paper, share-TV also provides communication services and reuse, however, this framework is limited to applications based on Google TV and Ginga-J platforms. The main difference is that share-TV is used to develop TV convergent applications while the work reported in this paper focuses on building smart TV applications synchronized with the content of TV programs.

Samsung Smart TV [11] presents a framework called AppsFramework. This framework encapsulates reusable modules for scene management, video playback / music, and so on. This makes it easier for the developer of smart TV applications to avoid performing complicated sequences of calls to the operating system in order to manage scenes (focusing, showing and hiding events) of an application, for instance. Some of these modules were used in the framework proposed in this paper.

Freitas and Teixeira [5] proposed an architecture for supporting the development of ubiquitous applications in home networks focusing on Digital TV. The proposed architecture consists of communication interface with home devices, a protocol layer for automatic service discovery, and so on. Although the architecture is designed to be implemented in iDTV middleware, some of its reusable artifacts such as the aforementioned ones were used and implemented in the framework proposed in this paper.

The framework for building synchronized smart TV applications with TV programs presented in this paper is different with all the aforementioned works from the scene that it uses image processing for synchronization purpose and it is a reference framework that allows more efficiency and less cost in building multi-platform applications in this field.

3 Synchronization

Teixeira et al. [13] studied the synchronization in the context of multimedia applications and considered that synchronization is a mechanism to guarantee that actions can happen according to the defined time reference set by a clock or established by the occurrence of events. It considers the tolerances that vary according to the type of application. In case of TV program, the characteristic of event is to be considered as a reference that is one of the important aspects of synchronization in the context of this study.

In order to provide synchronized applications with television programming, it is essential to know which channel the viewer is watching and what is being presented to the audience every moment. Then this information can be published to stakeholders and used to promote synchronization between program and application through notification services implemented in this work.

Notifications addressed in this work have various types such as channel identification, start and end of trading blocs, start, pause and end of the TV programs, sex scenes, violence, crime and some notifications generated by the events triggered by the user.

The synchronization module of this framework have API’s that allow applications to access notifications related to events occurring in TV programs. All these synchronization signals are generated by image and audio processing techniques. In this paper, only that synchronization mechanism is discussed which implemented by SURF, K-NN, K-Means and template matching techniques for channel identification.

3.1 Synchronization based on SURF

The main idea of this part is to, automatically, figure out what is the content of extracted frame from smart TV video. Although this problem of object detection and recognition in videos are seen as the frames sequence analysis, it can be simplified to single image analysis. Consequently, the main issue related to this task is how to compare two images (video frame and TV channel logo). We need to know if the logo image is present in the video frame. To do so, and after a research in the literature, we decide to use the SURF [7, 8] (Speed-Up Robust Features), an image features detector and descriptor inspired by SIFT [4] (Scale-Invariant Feature Transform). This decision took in consideration a precision and fast computation speed of SURF achieved by: (a) use of integral images for image convolutions; and (b) use of hessian matrix-based measure for detector and a distribution-based descriptor. Given an input image (video frame) and its extracted SURF features and corresponding descriptors, we need to match its descriptors with the logo images in the collection. With the huge number of logo descriptors in the collection, we need to avoid comparing with entire collection. For this, we use some segmentation techniques to divide all the features in sub-groups by common properties.

The process entails two sub-tasks: (1) data collection (Fig. 1(c)), a set of TV channel logos pick up from LyngSat [9] logo collection where for each logo class we defined the number of logo images and frames for training and testing phase; and (2) the real application (Fig. 1(e)). The first part contains the following tasks: (a) feature extraction: key-points are extracted from all logos (and all test frames) using SURF. These key-points are used for comparison; and (b) feature segmentation: receives large set of key-points as data features, extracted from TV channel logo collection, and segments all of them to obtain groups of common features.

The second part (Fig. 1(e)) is retrieving the logo(s) from an input image. This procedure has the following steps:

(a) features extraction: extracting the key-points of a video frame using SURF; (b) classifying the features: with all the features of a frame, we need to find which logo segment
matches the key-point better. All the logos point in those segments are possible candidates; (c) ranking the candidates: with all the candidates presenting some common key-points with the video frame, we need to rank these key-points by a determined threshold and keep only the similar ones; and finally (d) selecting the logo: after ranking the candidates, we have all the information to decide whether there is a logo inside the video frame or not and which one is contained.

3.1.1 Features Extraction
To extract features from each video frame, we used SURF algorithm implemented with OpenCV [3]. SURF is sufficiently fast for real time object recognition. For feature vector matching purpose, as showed in Fig. 3, we used K-Nearest Neighbors, Fast Library for Approximated Nearest Neighbors, distance ratio rule for finding all good matches and a threshold used to verify if the number of good matches found is enough to recognize a presence of the logo inside the frame. Some results of our experiments are shown in Fig. 4.

3.1.2 Collection Segmentation
After applying SURF algorithm and collecting a large dataset composed with TV logo vectors, we used K-Means clustering method to split this dataset in sub-groups containing common properties. K-means is a clustering method, widely used for partition problem, where given a dataset of n points, the objective is to divide this dataset in k groups finding k centroids. In this work, we partitioned the logo dataset in k vectors called centers or centroids. To obtain these centroids, we calculated the distance of all logo vectors to each k initial centroid during each iteration. Then, each logo vector belongs to its nearest centroid. We repeat this operation for a number of iterations where in each loop the groups are more balanced. By achieving the convergence, the process finishes. By convergence, we mean having no more change in the distribution of each group.

3.1.3 Classification and Ranking
The objective of this section is to use the K-Means and the relative segmented logos collection to classify the input video frame. To do so, we extracted the SURF vector from the frame and defined which the best group that contains elements more similar to the input SURF vector.

After applying the K-Means technique in the segmentation step, the result was a list of centroids that are, simply, the vectors which represent each group. Then, as it is shown in Fig. 6, we matched each frame SURF vector to all centroids in the K-Means model and we only kept the most similar one with the highest value of match.

Once we have these classes, all the candidates are ranked using the distance metric between the SURF points.

### 3.2 Synchronization based on template-matching technique
Multimedia synchronization, especially in smart TV environment, is tightly related to a synchronization of TV program video with smart TV applications. Through the image processing, anchors – associated with image transmitted by TV or parts of it - linked to the content can be defined. This technique allows realizing an image
processing, trying to find a similarity between images. Consequently, in the context of this study, it was possible to perform a more optimized search, trying to find points of synchronization inside the video frame, for example a TV channel logo, in our case.

With the extensive research on new and innovative techniques for multimedia indexing, actually we can find an OpenCV library, widely used in computer vision applications, which implements some of these algorithms. In this work, we used the CV_TM_CCORR_NORMED OpenCV method to compare smart TV video frame (query image) with the TV channel logo (reference image). This process is shown in Fig. 5.

4 Framework Architecture

Bosch et al. [2] report that framework development is different from a common application development. This is because of that framework's design needs to cover all relevant features of a particular domain and not just those ones of specific application. This is why it is important to consider the following when developing framework for smart TV integrated applications. Generally, there are six issues to consider. First, How to synchronize the TV content with Smart TV applications (Fig. 6(1)). Second, how mobile devices (smartphones, tablets, and others), which are in the same space with TV can interact with it (Fig. 6(2)). Third, how connected mobile devices in the TV environment can detect the presence of available TV service for use (Fig. 6(3)). Forth, how Smart TV applications can act over the TV controls such as changing channels, controlling volume and so on (Fig. 6(4)). Fifth, how ticker applications must share the remote control with the TV (Fig. 6(5)). Finally how a Smart TV application can identify the channel, which is being watched by the user (Fig. 6(6)). This last module was implemented using the TV channel logo detection presented in the section 3.

5 Framework Validation

To validate the framework proposed here we first developed a set of tools to implement test with developers and then used the experimental method proposed by Wohlin et al. [14]. The set is composed of two main parts: the Back-end (Fig. 7(a, b, c)) used to store the information of users and devices and to allow communication and sharing of media content among different components of the set; and the Front-end (Fig. 7(d, e)), which contains applications executed on mobile devices and Smart TV platforms.

The Back-end of this work provides web services for smart TV discovery services and for managing the users of social TV systems. The aforementioned services were developed using the Grails framework (Fig. 7(a)). Moreover, the Back-end allows communication and sharing the media content among the different applications.
respectively using Apache ActiveMQ message broker (Fig. 7(b)) and PHP server (Fig. 7(c)).

The Front-end is based on client-side (PC, smart TV, Tablet, smart Phone, etc.) that was developed using Apache Cordova + HTML5 + JQuery (Fig. 7(d)) and JavaScript-based smart TV framework API’s.

The objective of our experiment was:

- **To analyse** the use of the proposed framework in the construction of smart TV applications synchronized with television programing;
- **With the purpose** of evaluation
- **Regarding** the efficiency in terms of time spent and productivity;
- **From a point of view** of software developers;
- **In the context** of undergraduate and graduated in computer science and computer engineering. It is important to point out that in this experiment twelve (12) developers and one object were considered (TVMonitor: standard synchronized smart TV application).

The experiment consisted of a comparative study of the development processes of two versions of a standard synchronized smart TV application, one built with the reuse approach using the proposed framework and other built without this approach. We formulated three hypotheses and considered some metrics during the experiment.

For the formulation of the three hypotheses, the following metrics were considered:

- $\tau$: Total time spent by the team for developing smart TV application synchronized with the TV program;
- $\beta$: Team Productivity in terms of produced lines of code (LOC) per unit time ($\beta = \text{LOC} / \tau$);
- $\mu_1$: Average of the spent time by the teams for developing smart TV application synchronized with the TV program;
- $\mu_2$: Average productiveness of the teams in the development of smart TV application synchronized with the TV program.

We have a null hypothesis and its two corresponding alternatives:

- **Null Hypothesis (H0):** There is no difference between teams who used the proposed framework and teams that did not use while developing TVMonitor application regarding the efficiency ($\epsilon$) of the team.
  
  $H_0$: $\epsilon_{\text{framework}} = \epsilon_{\text{withoutframework}} = \mu_{\text{framework}} = \mu_{\text{withoutframework}}$

- **Alternative Hypothesis (H1):** Teams who use the proposed framework for building TVMonitor application are generally more efficient than those developing with the use of framework.
  
  $H_1$: $\epsilon_{\text{framework}} < \epsilon_{\text{withoutframework}} \Rightarrow \mu_{\text{framework}} > \mu_{\text{withoutframework}}$

- **Alternative Hypothesis (H2):** Teams using the approach “without framework” for building TVMonitor application are generally more efficient than those developing with the use of framework.
  
  $H_2$: $\epsilon_{\text{framework}} > \epsilon_{\text{withoutframework}} \Rightarrow \mu_{\text{framework}} < \mu_{\text{withoutframework}}$

The organization of the data collected during the experiment in Fig. 8 is done according to the two development approaches used in this experiment: development with and without the use of the framework reported in this paper.

An initial analysis was done on the data collected in Fig. 8. It is important to note that the distribution efforts of the groups in the design and test phases of TVMonitor development was constant. However, there is a great discrepancy of development efforts for the groups that used the proposed framework during the implementation of TVMonitor. While groups which have not used the proposed framework, in average they spent 4 hours and 59 minutes but those who used the framework spent 2 hours and 08 minutes (a decrease of 57, 2%).

Finally, we tested our hypotheses using the t-test [10] which aims to verify that a variable differs between two independente samples, based on the arithmetic average and considering variability of its data items. Then with some degree of significance ($\alpha$), reject the null hypothesis ($H_0$) and choose one of the alternative hypothesis ($H_1$ or $H_2$). The t-test formula is given by equation (1), where $Sx^2$ and $Sy^2$ are the variances of each sample; $Sp$ is the dispersion; and $n$ and $m$ are the numbers of data items that each sample contains. In equation (2), $n+m-2$, typically noted by gl is called the degree of test’s freedom.

\[
\begin{align*}
t_0 &= \frac{x - \bar{y}}{Sp \sqrt{\frac{1}{n} + \frac{1}{m}}} \\
Sp &= \sqrt{\frac{(n-1)Sx^2 + (m-1)Sy^2}{n+m-2}}
\end{align*}
\]

Once you have calculated $t_0$, $\alpha$ and gl, you can check the value of the standard $t$ in t-test distribution to see if $t_0$ is so significant.

If $|t_0| >$ standard $t = t_{\alpha/2, gl}$ → REJECT $H_0$, Otherwise, $H_0$ NOT REJECTED and no conclusion is drawn from the experiment.

As the dependent variable of the experiment (efficiency of teams) has two treatments (total time ($\tau$) and productivity...
(b)), the application of t-test was performed in two steps too. During this process, we calculated the variance using the following equation:

\[ S_x^2 = \frac{\sum_{j=1}^{n}(x_j-\bar{x})^2}{n-1} \] (3), with \( n = 3 \)

**Step 1: t-test (Total Time)**

After calculating the variance of each group, we have:

\[ S_x^2 \text{ (without framework)} = 2.97723 \]
\[ S_x^2 \text{ (with framework)} = 1.5325 \]
\[ \alpha = 0.2 \]
\[ t_{0.02, 4} = t_{0.100, 4} = 2.1318 \]
\[ t_0 = -2.498498775014366 \]

Then we have \( |t_0| > t_{0.02, 4} \) REJECT the null hypothesis \( H_0 \) with 20% of significance.

**Step 2: t-test (Total Productivity)**

After calculating the variance of each group, we have:

\[ S_x^2 \text{ (without framework)} = 15951 \]
\[ S_x^2 \text{ (with framework)} = 179347 \]
\[ \alpha = 0.02 \]
\[ t_{0.02, 4} = t_{0.01, 4} = 4.6041 \]
\[ t_0 = 5.475964737075994 \]

Then we have \( |t_0| > t_{0.01, 4} \) REJECT the null hypothesis \( H_0 \) with 2% of significance.

**6 Final Remarks**

The proposed framework can be offered to developers in a semi-complete source code skeleton that integrates synchronization, notification and TV controls functions. A set of tools that were developed in section 5 can provide to a developer more facilities in the process of building Smart TV synchronized applications. In addition, the experiment conducted in this paper could prove statistically that the proposed framework can be considered as an important tool to support the developers of applications in this field. In addition, all functional requirements that were established while planning the development of this framework were implemented and used during the instantiation of the proposed framework.

Regarding future work, we plan to: (a) add mechanisms of synchronization through local audio/video processing of TV content to framework; (b) explore and add adjustment mechanisms of synchronization with the purpose of minimizing the delay difference that exists among various forms of television content transmission (radio broadcasting, cable, satellite, etc.); and (c) offer more notification API’s for supporting the development of social TV systems with the purpose of improving the user experience quality in Smart TV environment.

**7 References**


