Mechanism to Validate and Compare Inter-Device/Destination Media Synchronization Solutions Adopted in Multi-Screen Scenarios

D. Marfil, F. Boronat, *Senior, IEEE*, J. López, B. Roig

***Abstract*— Simultaneous consumption of related content in one (or multiple) devices (i.e., multi-screen scenarios) is commonplace. To provide a satisfactory Quality of Experience (QoE) in this type of scenarios it is necessary to adopt synchronization mechanisms to keep the involved content playout processes in a synchronized state. Most of those mechanisms are software-based and usually perform some estimations, at runtime, of the playout latency. Therefore, the accuracy of the measured synchronization level can be difficult to evaluate or assess. In this work, a computer vision-based mechanism to evaluate the achieved accuracy in multi-screen scenarios is proposed. Thus, the accuracy level which is obtained through the synchronization mechanisms implemented in those systems or applications can be assessed at the end point of the content presentation. Additionally, in order to show the utility of the proposed mechanism, some evaluations have been conducted for a common use case, a videowall, in which accurate synchronization is a requirement.**

***Index Terms*—computer vision, latency, multimedia synchronization, OCR, image recognition.**

I. Introducción

Actualmente existe una gran proliferación de aplicaciones que requieren el uso de múltiples pantallas donde se reproducen contenidos relacionados de forma sincronizada en múltiples escenarios. Como ejemplo, se pueden citar: *videowalls* publicitarios o paneles informativos, aplicaciones de segundas pantallas[1] relacionadas con la TV, etc. En muchos casos, estas aplicaciones permiten que el usuario pueda consumir contenidos multimedia de manera simultánea en los diferentes dispositivos involucrados. Sin embargo, estos contenidos pueden ser heterogéneos (diferente codificación, formato, etc.), recibidos a través de redes de distribución heterogéneas (p.ej., *broadcast* o *broadband*) y consumidos en una gran variedad de dispositivos de consumo, con especificaciones y rendimientos heterogéneos.

Submitted for review: 19/11/2019. This work has been funded by the Generalitat Valenciana, Investigación Competitiva Proyectos, through the R&D Program "Grants for research groups to be consolidated, AICO/2017" under Project AICO/2017/059.

D. Marfil, F. Boronat, J. López and B. Roig are with the Universitat Politècnica de València (UPV), Campus de Gandia, 46730 Gandia (Spain) (e-mails: damarre@ dcom.upv.es; fboronat@ dcom.upv.es; jailogu@ epsg.upv.es; and broig@ upv.es).

[1] https://www.emarketer.com/content/millennials-favor-smartphones-for-second-screening

Todo ello implica que el retardo extremo-a-extremo entre el origen de los contenidos y su consumo por parte del usuario puede variar dependiendo del tipo de contenido, red utilizada y dispositivo de consumo. Como ejemplo, en la Fig. 1 se puede observar la variabilidad existente en el retardo extremo-a-extremo cuando se ven involucrados diferentes flujos y según la tecnología utilizada debido a los diferentes elementos existentes en cada cadena de distribución.

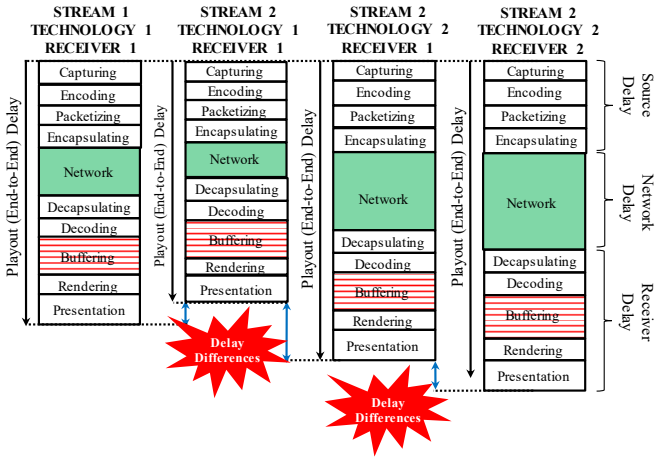

Fig. 1. Variabilidad de retardo.

Por tanto, con el fin de que el usuario final pueda consumir múltiples contenidos (relacionados) de forma simultánea y sincronizada, deben establecerse mecanismos que permitan ajustar los procesos de reproducción en cada uno de los dispositivos involucrados, para que, de esta manera, presenten al usuario, de forma sincronizada, fotogramas que hayan sido grabados o generados en el mismo instante de tiempo, bien por un mismo dispositivo de captura o bien por varios dispositivos.

De hecho, en [1] se presentan los resultados de un estudio centrado en las preferencias, hábitos de consumo y las expectativas de más de 1000 usuarios españoles con relación a servicios de TV híbridos (con consumo de contenidos recibidos por redes *broadcast* –DVB-T- y *broadband* –redes IP-). En dicho estudio se identificó el problema de la sincronización como uno de los más importantes a la hora de proporcionar una calidad de experiencia satisfactoria a los usuarios finales.

Respecto a la sincronización, en [2] se definen los diferentes tipos de sincronización existentes. La sincronización intra-flujo facilita que, dentro del mismo flujo, la información discurra de manera ordenada (p.ej., presentar ordenadamente y

equiespaciados en el tiempo los fotogramas en un flujo de vídeo, para una visualización coherente). La sincronización inter-flujo facilita que en un contenido los diferentes flujos involucrados estén presentados y asociados correctamente (p.ej., que el audio y la imagen de un contenido estén reproduciéndose en paralelo de forma coherente). La sincronización inter-dispositivo (*Inter DEvice Synchronisation* o IDES), facilita que varios dispositivos independientes (pero físicamente cerca) reproduzcan, de forma simultánea, el mismo contenido o contenidos relacionados. Finalmente, la sincronización inter-destinatario (*Inter Destination Media Synchronisation* o IDMS), facilita que varios dispositivos en destinos separados geográficamente reproduzcan de forma sincronizada el mismo contenido o contenidos relacionados. Generalmente, los mecanismos de sincronización intra- e inter-flujo ya están implementados en cualquier reproductor y suelen funcionar bien. En este trabajo se presenta una herramienta diseñada para calcular la precisión de sincronismo alcanzado en entornos IDES, aunque también serviría para entornos IDMS simulados en laboratorio.

Normalmente, los mecanismos de sincronización son módulos software que realizan cálculos y estimaciones del retardo de reproducción en el momento de la recepción o la decodificación del contenido. En dichos instantes, aún existe una latencia hasta la presentación del contenido en pantalla (momento en el cual el usuario visualiza dicho contenido) que, a priori, es desconocida y, por tanto, debe ser estimada por dichos mecanismos (Fig. 2).

Este tipo de mecanismos se han utilizado por los autores, por ejemplo, en [3]. Aunque los resultados relativos al nivel de sincronismo alcanzado que se exponen en dicho trabajo son satisfactorios (tanto objetiva como subjetivamente), al basarse en estimaciones de la latencia hasta el momento de la presentación de los contenidos en pantalla, se desconoce su precisión exacta.

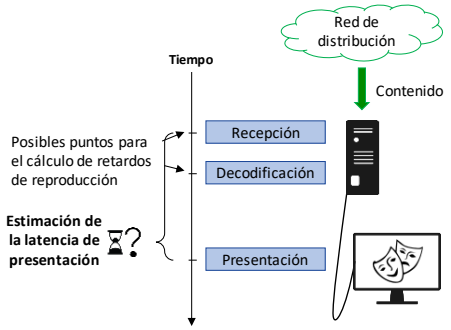

Fig. 2. Puntos en el proceso de reproducción en los que normalmente se calculan los valores de asincronía.

Es por ello, que se necesitan otros mecanismos de cálculo de asincronías que proporcionen unas medidas más exactas (es decir, no basados en estimaciones) del grado de sincronización obtenido, con el fin de obtener resultados lo más precisos posibles. En este artículo se presenta un mecanismo basado en técnicas de visión artificial que permite realizar medidas del nivel de sincronismo alcanzado en el momento de la visualización del contenido en la pantalla, es decir, en el instante de la presentación del contenido al usuario (Fig. 3). Para ello, se elegirá contenido lo más realista posible (esto es, contenido multimedia con cambios de escenas, movimiento, etc., evitando utilizar contenidos similares a las cartas de ajuste) y se adaptará para que incluya cierta información visual que, a través de herramientas de visión artificial, pueda ser reconocida e interpretada correctamente. A cada fotograma del vídeo se le deberá insertar, para que se visualice de forma superpuesta, información relacionada con el *timing* del vídeo, como, por ejemplo, el instante de generación del mismo o una cadena de texto incluyendo el número de dicho fotograma dentro de la secuencia de vídeo. De esta manera, es posible calcular el valor de asincronía existente entre las presentaciones de los contenidos en las diferentes pantallas de los dispositivos involucrados, comparando dicha información visual superpuesta con la detectada por el sistema al reproducirse el contenido en cada una de dichas pantallas.

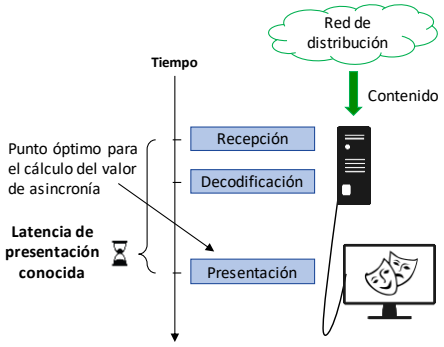

Fig. 3. Punto óptimo en el proceso de reproducción para calcular los valores de asincronía.

Esto se puede realizar de forma automatizada utilizando las numerosas técnicas y mecanismos para el tratamiento y procesado de imágenes existentes hoy en día (p.ej., desde filtros hasta técnicas más complejas para reconocer objetos, etc.). En concreto, las técnicas de reconocimiento óptico de caracteres (*Optical Character Recognition*, OCR) son mecanismos que permiten detectar cadenas de caracteres que puedan existir en imágenes o fotografías. Dichas técnicas permiten interpretar el texto de una forma mucho más eficiente, siendo su aplicabilidad muy variada (p.ej., desde transcribir un libro físico de una manera rápida y automatizada hasta detectar y almacenar la matrícula de un vehículo entrando a un garaje o que haya excedido los límites de velocidad).

El mecanismo descrito en este artículo presenta las siguientes características:

- Realiza el cálculo de las asincronías en instantes de reproducción (más fiable) y no en instantes anteriores como los métodos tradicionales (basados en estimaciones y, por tanto, menos fiables). Permite realizar un cálculo más preciso y fiable del grado de asincronía alcanzado entre dispositivos, en aquellos casos en los que los mecanismos utilizados no proporcionen una precisión satisfactoria o cuyas estimaciones realizadas no sean fiables.
- Está basado en técnicas de OCR para medir la asincronía en la presentación de contenidos relacionados cuando son mostrados en diferentes pantallas conectadas a uno o varios dispositivos.
- Permite realizar una comparativa con el valor de asincronía que los dispositivos involucrados han estimado que existía en cada momento al realizar los ajustes

correspondientes en los procesos de reproducción. Con esto, se puede comprobar (y validar) el nivel de precisión alcanzado con dichas estimaciones y, en caso de que dicho nivel no sea suficiente, realizar los ajustes correspondientes con el fin de mejorar el mecanismo de sincronización empleado (p.ej., corregir o ajustar los parámetros utilizados en los mecanismos encargados de estimar la latencia asociada al proceso de reproducción).

Este artículo es una extensión del artículo presentado en las Jornadas de Ingeniería Telemática (JITEL) en octubre de 2019 [4]. En dicho artículo se define el mecanismo de una forma muy breve, esquemática y mucho más generalista. Además, se aplica sobre un caso de uso más sencillo, omitiendo una serie de resultados numéricos y gráficos que sí se detallan en este artículo y permiten concluir, experimentalmente, que el mecanismo propuesto funciona satisfactoriamente. Las contribuciones y diferencias respecto a dicho artículo se enumeran en la siguiente lista:

- Un mayor detalle en la descripción, el contexto de la utilidad y las contribuciones del mecanismo propuesto.
- Se presenta la versión definitiva del mecanismo, así como una validación de su utilidad. En [4] se presentó una versión preliminar del mecanismo, aún en proceso de depuración (*work-in-progress*).
- Validación más exhaustiva del mecanismo propuesto (sección IV). Se incluyen más medidas, resultados, incluso gráficas y figuras de especial relevancia que no aparecían en [4]. Además, dichos resultados se han obtenido en un caso de uso de mayor complejidad al considerado en [4]. Se han realizado más medidas y, por tanto, los resultados proporcionados son más fiables, con medidas más rigurosas (p. ej. se proporciona la asincronía detectada ± intervalo de confianza del 95%).

El artículo sigue la siguiente estructura: en la Sección II se hace referencia al estado del arte y trabajos existentes relacionados, tanto con técnicas de sincronización o cálculo de latencias en sistemas inter-dispositivo/destinatario a partir del procesado de imágenes, como con técnicas de tratamiento y procesado de imagen OCR. En la Sección III, se presenta el mecanismo basado en técnicas OCR propuesto para el cálculo de precisión de la sincronización alcanzada entre dispositivos. En la Sección IV, se presenta un caso de uso para la valoración y validación del mecanismo de sincronización utilizado en un sistema propio de *videowall*. Finalmente, en la Sección V, se exponen las conclusiones sobre la utilidad del mecanismo propuesto y los resultados obtenidos, así como posible trabajo relacionado a realizar en el futuro.

II. Estado del Arte

En esta sección se resumirán los principales trabajos existentes relacionados con el mecanismo propuesto en este artículo. En primer lugar, se presentan trabajos relacionados con el cálculo del nivel de sincronización (o latencia) existente a través de mecanismos o herramientas de procesado de imagen (subsección II.A). A continuación, se presentan trabajos específicos relacionados con técnicas basadas en OCR, que ha sido la técnica adoptada para el análisis de las imágenes capturadas para el cálculo de la sincronización alcanzada (subsección II.B). Finalmente, se incluye una breve discusión sobre las contribuciones y la finalidad del mecanismo propuesto (subsección II.C).

A. Mecanismos de Evaluación de Sincronización Basados en el Procesado de Imágenes

Respecto a la adopción de técnicas de procesado de imagen en este campo, en [5] se propone una solución para el cálculo de retardos entre los diferentes dispositivos involucrados en una videoconferencia. Dicha solución consiste en la generación y análisis de imágenes detectables por ordenador, concretamente códigos QR con información temporal. Estas imágenes son analizadas por otros equipos con dispositivos de entrada de vídeo para poder calcular el retardo existente en la videoconferencia. El procedimiento para calcular dicho retardo puede resultar incómodo, pues para poder medirlo, los extremos (es decir, los participantes en la videoconferencia), deben apuntar con su webcam a la pantalla de sus equipos, que es donde estarán visualizándose los códigos QR (el del propio usuario y el del usuario remoto). De esta forma, el mecanismo que se propone en dicho trabajo puede detectar ambos códigos y obtener la diferencia de tiempos entre ambos (retardo extremo a extremo). El prototipo que se implementa en dicho artículo sólo es compatible con MacOSX y, por consiguiente, no es multiplataforma. Además, requiere de una calibración previa para eliminar el retardo inducido por el propio procesado de imagen que realiza dicho mecanismo.

En [6], se presenta un mecanismo para evaluar, a través del análisis de imágenes y de audio, el nivel de sincronismo alcanzado utilizando una solución de sincronización inter-dispositivo (IDES) basada en el estándar DVB-CSS (Digital Video Broadcasting - Companion Screens and Streams [7]). Para tal fin, en ese trabajo se proporciona un vídeo que incluye ciertos pitidos y flashes en instantes conocidos. Este tipo de eventos audiovisuales son recogidos por un microcontrolador Arduino Due [8], el cual está conectado por USB a un PC con el rol de uno de los dos dispositivos (pantalla principal -main screen-o pantalla complementaria -*Companion Screen*-). El cálculo de los retardos (asincronías) se calcula obteniendo la diferencia entre los instantes de tiempo en los que se espera que haya pitidos o flashes en el dispositivo implementado en dicho PC y los instantes de tiempo en los que se detecta que ocurren dichos eventos en el otro dispositivo.

Por otro lado, en [9] se presenta un mecanismo para el cálculo de retardos de vídeo extremo-a-extremo. Dicho mecanismo consiste en la inserción en cada fotograma de una marca temporal codificada en formato de código de barras. En ese trabajo se utiliza el *framework* GStreamer [10], por lo que la generación y posterior detección de las marcas de tiempo insertadas se lleva a cabo dentro del propio proceso de reproducción, a través de un elemento de GStreamer implementado para dicha finalidad (en concreto, el elemento *videodetect* [11]).

B. Técnicas de Reconocimiento de Caracteres OCR

Las técnicas OCR, de reconocimiento de caracteres, llevan investigándose desde hace décadas. Ya en 1990, en [12] se recopiló una gran variedad de mecanismos existentes para el reconocimiento de diferentes formatos de caracteres (p.ej., para una o más fuentes de texto específicas, para texto escrito a mano, etc.). De acuerdo con dicho trabajo, según la manera de

analizar la imagen, las técnicas OCR pueden clasificarse en dos: 1) a través de la utilización de plantillas, en las que el texto a analizar se compara con unos prototipos de caracteres previamente almacenados; y 2) a través del análisis de los parámetros del carácter a reconocer y técnicas de emparejamiento (*matching techniques*). Cabe resaltar que la segunda opción es la más utilizada y consiste, principalmente, en la extracción de parámetros significativos del carácter analizado y su posterior comparación con parámetros de caracteres ideales. Tras esta comparación, se asume que el carácter ha sido reconocido cuando sus parámetros son muy similares a los de uno de los caracteres ideales. El grado de similitud obtenido proporciona el nivel de confianza con el que se ha interpretado el carácter.

Actualmente, también se emplean técnicas más avanzadas para el reconocimiento de texto. Como ejemplo, el trabajo en [13] describe el reconocimiento de escritos a mano mediante el uso de redes neuronales. En dicho trabajo, los caracteres se redimensionan en áreas de 60x40 píxeles y son introducidos en la red neuronal. Cabe destacar que, en dicho trabajo, la red neuronal ha sido entrenada para el alfabeto y lenguaje inglés con más de 19.000 muestras, alcanzando una precisión del 95,69%.

Para el reconocimiento de caracteres, existen numerosas librerías (muchas de carácter *open-source*) que pueden implementarse en diferentes lenguajes de programación, tales como, por ejemplo, python [14] o javascript (nodejs [15]), así como en otros entornos como Matlab [16], que es una herramienta de cómputo numérico con un lenguaje de programación propio.

C. Discusión y Contribuciones del Mecanismo Propuesto

Tal y como se ha visto en la subsección II.A, a pesar de existir técnicas que permiten calcular el nivel de sincronización (o latencia) entre los dispositivos involucrados en escenarios o sistemas multi-pantalla, dichos trabajos previos requieren de una complejidad extra o están diseñados con el fin de ser utilizados en aplicaciones muy específicas (p. ej., videocoferencias en [5] o TV en [6]). Es por ello que resulta necesario un mecanismo agnóstico que pueda ser utilizado para validar este tipo de sistemas sin añadir una dificultad extra a su instalación y que, además, pueda aplicarse en cualquier sistema cuyo mecanismo de sincronización requiera ser validado. A diferencia de otros mecanismos existentes, el mecanismo propuesto en este artículo permite su aplicabilidad en todo tipo de sistemas o escenarios multi-pantalla y se basa en técnicas poco complejas, ampliamente utilizadas y optimizadas (como, por ejemplo, el procesado de imágenes y la detección óptica de caracteres).

III. Mecanismo para el Cálculo del Sincronismo Alcanzado en Entornos Multi-Dispositivo

En esta Sección, se presenta un mecanismo no intrusivo para el cálculo, mediante técnicas de visión artificial, del nivel de sincronización adquirido en el instante de presentación de varios dispositivos que deben estar reproduciendo el mismo contenido, o bien contenidos relacionados, de forma sincronizada. Los autores consideran como mecanismo no intrusivo aquel que puede ejecutarse sin formar parte de la aplicación o plataforma objeto de estudio. Por el contrario, un mecanismo intrusivo podría basarse, por ejemplo, en modificar una aplicación para generar una serie de registros en un determinado fichero, modificando así la priorización y el uso de recursos de los dispositivos por parte de la propia aplicación y, por tanto, pudiendo afectar a su rendimiento global.

Este mecanismo es capaz de obtener, a partir de un dispositivo de entrada de vídeo (p.ej., una webcam), información temporal de cada fotograma que está siendo presentado en cada uno de los dispositivos involucrados. Para ello, el contenido utilizado deberá ser preparado para incluir dicha información de forma superpuesta (*overlay*) para que pueda ser detectada por dicho dispositivo (explicado en la subsección III.A con mayor detalle). En este artículo, se propone utilizar los números de cada fotograma superpuestos en cada uno de ellos. Dicha información que se analiza a partir de imágenes obtenidas mediante una cámara de vídeo, junto con la información relativa a la tasa de fotogramas por segundo (fps) del contenido, permite calcular el valor de la asincronía máxima existente entre los dispositivos. La cantidad máxima de imágenes que podrán ser obtenidas por segundo dependerá del dispositivo de captura empleado. Dichas imágenes podrán ser procesadas en tiempo real o a posteriori, dependiendo de la capacidad de procesamiento del HW empleado.

Se puede calcular el nivel de asincronía existente en cada instante entre los *N* dispositivos involucrados de la siguiente manera (Ec. 1): dado un instante de tiempo *t* (en segundos), en el que se realiza la medida (captura de la imagen), la asincronía máxima para dicho instante será la diferencia entre los números de fotogramas máximo y mínimo que se estén mostrando en dicho instante en las pantallas de los dispositivos ($\max(n_{trama})$, $\min(n_{trama})$, respectivamente) multiplicada por la duración de un fotograma de dicho contenido ($\frac{1}{fps}$, siendo *fps* la tasa del vídeo en fotogramas por segundo).

$$Asincronía_{max}(t) = (\max(n_{trama}, t) - \min(n_{trama}, t)) * \frac{1}{fps}\ [segundos] \quad (1)$$

Cabe señalar que existe un margen de error en el resultado que se obtiene a partir de la Ec. 1 debido a que no se pueden medir asincronías con un valor menor que la correspondiente a la duración de un fotograma. Dicho error, denominado Error de Precisión (EP), se puede calcular (en segundos) a partir de la Ec. 2 de la siguiente forma:

$$EP = \pm \frac{1}{fps}\ [segundos] \quad (2)$$

Por tanto, si tras realizarse los cálculos pertinentes, se obtiene que la diferencia entre números de fotogramas de los dispositivos respecto a la referencia es de *n* fotogramas, esto implicaría un nivel de asincronía (en segundos) cuyo valor se encuentra dentro del intervalo definido en la Ec. 3:

$$Asincronía \in \left\{\frac{n}{fps} - EP, \frac{n}{fps} + EP\right\}[segundos] \quad (3)$$

A pesar del EP, el mecanismo puede considerarse suficiente, al ser éste una manera de evaluar la precisión de sincronismo alcanzada en los sistemas evaluados. Es decir, de evaluar y validar el comportamiento de las técnicas de sincronización que ya están implementadas en las aplicaciones y los dispositivos objetos de la evaluación.

Resumiendo, el mecanismo de medida propuesto se puede dividir en dos fases: 1) la preparación del contenido; y 2) la medición de asincronías en tiempos de presentación.

A. Fase 1: Preparación del Contenido

Se debe utilizar un contenido que disponga de marcas o referencias superpuestas, elementos identificables por el mecanismo propuesto. Se puede utilizar cualquier tipo de contenido siempre y cuando haya un proceso previo encargado de la inserción de dicha información. Debido a esto, el contenido generado específicamente para el mecanismo propuesto no debería ser utilizado por usuarios, ya que contará con elementos extraños (*artifacts*), como las cadenas de texto superpuestas, tapando parte del contenido, que pueden resultar molestos y empeorar la calidad de la experiencia de consumo.

Una manera simple de incluir dicha información identificable fácilmente es superponer el número de fotograma asociado a cada fotograma de vídeo. Entre muchas otras, una de las herramientas que lo permite de forma muy sencilla es *ffmpeg* [17]. En la Fig. 4 se muestra el comando *ffmpeg* para insertar el número de fotograma en un vídeo.

```
ffmpeg -i {contenido_original} -filter_complex "drawtext=
fontfile=/usr/share/fonts/truetype/freefont/FreeSerif.ttf: text='frame %{n}':
x=100: y=50: fontsize=80: fontcolor=white@1.0: box=1:
boxcolor=black@1.0" {contenido_generado}
```

Fig. 4. Comando *ffmpeg* utilizado para la inserción del número de fotograma.

Donde *{contenido_original}* es la ubicación y nombre del contenido al que se le va a superponer el número de fotograma de vídeo, *{contenido_generado}* es la ubicación y nombre del fichero generado con los números de fotograma superpuestos. Se utiliza el parámetro *filter_complex* para indicar qué se va a superponer en el vídeo. En el ejemplo, se está insertando el texto "frame *n*", siendo *n* el número de fotograma y la palabra *frame* como identificador de que el número que acompañe a dicha cadena de caracteres corresponde con el número de trama. Los autores consideran como muy poco probable que un contenido audiovisual vaya a presentar de forma original esta palabra. Además, se utilizan otros parámetros que permiten configurar la posición donde se insertará este texto (variables *x* e *y,* en el comando), o el tamaño y color del texto y si debe tener un fondo (variables *fontsize*, *fontcolor*, *box* y *boxcolor*, respectivamente). La Fig. 5 muestra cómo quedaría el resultado final, tras la generación del contenido a partir del comando de la Fig. 4. Se puede apreciar que se ha superpuesto la cadena de texto "frame 936" en color blanco sobre un fondo negro. Dicho formato condicionará los filtros de imagen a aplicar en el sistema, tal y como se explica más adelante.

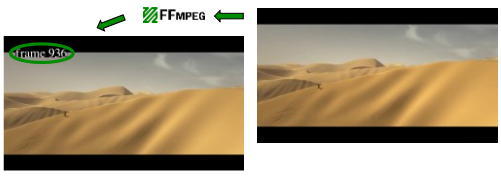

Fig. 5. Contenido con el número de fotograma insertado a través de la herramienta *ffmpeg*.

B. Fase 2: Medición de Asincronías en Tiempos de Presentación

Una vez que el contenido ya cuenta con el número de fotograma superpuesto, se lleva a cabo un calibrado manual, previo al inicio de la evaluación del sistema, con el fin de situar el dispositivo de captura de imagen correctamente y determinar la sección a recortar de la imagen obtenida, ya que, dependiendo del dispositivo de captura de imagen empleado, la distancia al sistema a evaluar y la sección a recortar de la imagen obtenida puede variar (esto es, depende directamente de la óptica y la resolución del dispositivo de captura de imagen utilizado). Tras finalizar el calibrado, se iniciará el sistema objeto de evaluación, y, por tanto, la reproducción de dicho contenido en los dispositivos involucrados. A partir de dicho instante, ya se puede empezar a capturar vídeo o imágenes de forma periódica, y medir y comparar los valores de los fotogramas que están siendo presentados en las pantallas de cada uno de los dispositivos involucrados.

Dependiendo del rendimiento del dispositivo que esté a cargo del análisis de las imágenes, podrá realizarse la medida de la asincronía en tiempo real, o bien se podrán almacenar las imágenes capturadas de la entrada de vídeo para su análisis posterior. Por ejemplo, la Fig. 6 muestra un esquema de los posibles dispositivos involucrados para el mecanismo propuesto, donde se puede observar un PC con una entrada de vídeo (una cámara) conectada. Esta cámara captura la imagen de las pantallas involucradas en el sistema para el que se quiere medir el nivel de sincronismo. El PC al que está conectada la cámara se encarga de analizar las imágenes obtenidas y, a continuación, calcular dicho nivel alcanzado entre los *n* dispositivos del sistema.

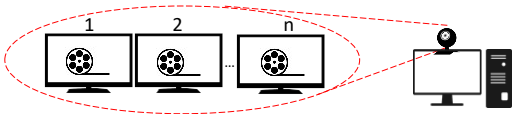

Fig. 6. Esquema de los dispositivos involucrados para el cálculo del nivel de sincronismo alcanzado.

El nivel de sincronismo alcanzado se evalúa de la forma siguiente: mientras las pantallas del sistema a evaluar están reproduciendo el contenido preparado en la fase 1, el sistema de medida obtiene periódicamente, mediante capturas de imágenes por la cámara y su procesado mediante software, los números de fotograma mostrados en cada pantalla y calcula y registra la asincronía máxima entre las pantallas en el instante de cada captura mediante la Ec. 1.

Con el objetivo de agilizar el procesado de la imagen a evaluar, se realiza un recorte en la imagen captada para almacenar solamente las regiones de interés de cada pantalla (donde están visibles los números de fotograma). Seguidamente, se aplican una serie de filtros que permiten descartar aquellas regiones que no van a aportar información.

Los filtros adoptados para el procesado de imagen, previo al proceso de detección de caracteres, son los típicamente utilizados en técnicas OCR, como el filtro *top-hat* [18] o el filtro de *erosión* [19], los cuales permiten destacar los números de fotogramas superpuestos y ocultar el resto de información de la imagen. Concretamente, el filtro *top-hat* permite resaltar regiones más claras respecto a las oscuras, por lo que, permitirá destacar la zona en la que se encuentra la información del número de fotograma (al ser el texto de color blanco sobre fondo negro). El filtro de *erosión* permite "limpiar" la imagen eliminando elementos diferentes a un tamaño configurado. Esto permite eliminar zonas que puedan dar lugar a falsos

positivos, es decir, evitar la detección de caracteres donde no los hay. Tras estos pasos, la imagen procesada ya estará preparada para utilizar el mecanismo OCR con el fin de detectar los números de fotograma que existan en la misma. Finalmente, tras obtener la información de los números de fotograma, se podrá hacer uso de la Ec. 1, y obtener la asincronía máxima existente para cada instante. Como resumen, la Fig. 7 muestra las distintas etapas por las que pasa la imagen durante el proceso de detección de la información.

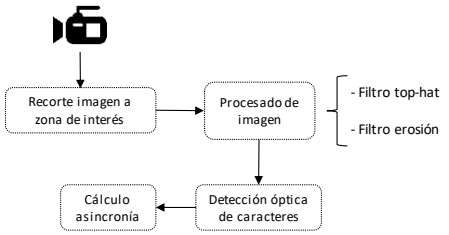

Fig. 7. Distintas etapas del tratamiento de la imagen para detectar la asincronía máxima entre dispositivos de un sistema.

IV. Validación del mecanismo propuesto: Caso de uso

En esta Sección se va a utilizar el mecanismo propuesto para evaluar y validar el mecanismo de sincronización implementado en un caso de uso de un sistema propio de pantallas múltiples, formando un *videowall*, basado en dispositivos Raspberry Pi (en adelante, RPi), cuya descripción detallada puede encontrarse en [20]. En concreto, para esta evaluación se ha utilizado un sistema *videowall* de 2x3, es decir, compuesto por 6 pantallas distribuidas en dos filas y tres columnas. El mecanismo de sincronización del *videowall* ha sido realizado por los autores y se basa en un sistema de control Maestro/Esclavo y en el cálculo y estimación de asincronías[2] realizados por el software de reproducción, ejecutándose en las propias RPi, en el momento de la recepción del contenido. En ese momento, se realiza una estimación de la latencia asociada al proceso reproductor desde dichos instantes hasta la presentación del contenido en la pantalla. El sistema propuesto en este artículo nos permitirá, además, comprobar el nivel de precisión alcanzado en el sistema de *videowall* bajo estudio. Los dispositivos involucrados para obtener las imágenes y calcular la asincronía se muestran en la Tabla I. La cámara estaba colocada delante del videowall, a 2m del mismo.

El contenido utilizado ha sido el tráiler del vídeo Sintel [21]. Se ha seleccionado este vídeo, y no un vídeo más uniforme (p.ej., una carta de ajuste), porque se desea evaluar la sincronización con vídeos que sean realistas, es decir, que sean similares a los que se pueden encontrar en casos reales en un *videowall* (con cambios de escena, movimientos, etc., que puedan afectar al uso de recursos y de capacidad de procesamiento en los dispositivos que ejecutan los procesos de sincronización). Los parámetros y características del contenido seleccionado están listadas en la Tabla II.

Puesto que en el *videowall* se visualizan diferentes partes del contenido en cada pantalla, mediante ffmpeg, se ha realizado la superposición del número de fotograma en 6 puntos diferentes para que en todo momento se visualice esta información en todas las pantallas (ver Fig. 8).

[2] La descripción del mecanismo de sincronización implementado y de cómo se realizan las estimaciones del retardo de reproducción no son objeto de este artículo.

TABLA I.
Dispositivos Utilizados

PC	Windows 10, i7 6700 @ 3.40GHz, 8GB RAM, 1TB HDD.
Webcam	Logitech HD C270 (720p, 30fps).

TABLA II.
Parámetros del Contenido Utilizado

Codificación	H.264 + AAC
Resolución	1920x818 px
Fotogramas por segundo	25 fps (i.e., 40 ms por fotograma)
Duración	52 s

Fig. 8. Contenido preparado con el número de fotograma superpuesto.

Para poder comparar los valores medios de asincronía obtenidos a partir del software del propio sistema *videowall* con los que se obtienen a partir del mecanismo propuesto en este artículo, se han llevado a cabo 10 sesiones de aproximadamente 5 minutos cada una (con el vídeo reproduciéndose en bucle), con el objetivo de obtener el valor medio de sincronismo alcanzado según el software del *videowall* (es decir, durante la recepción del contenido y estimando la latencia de los procesos de reproducción).

Tras analizar los valores almacenados por el software de sincronización del *videowall*, se ha obtenido un valor de asincronía media de 33ms con un intervalo de confianza del 95% (I.C. 95%) de ±3.9ms, siendo el tiempo entre fotogramas de 40ms. Por tanto, si dichos valores han sido estimados correctamente, el resultado que se debería obtener a través del mecanismo presentado en este artículo debe ser un valor medio que esté entre 0ms ± $\frac{1}{fps}$ ms. Para este caso específico será un valor entre 0ms y ±40ms (Ec. 2 y 3). Ello implica que, durante el análisis con el mecanismo propuesto, debería obtenerse de promedio hasta un fotograma de diferencia entre los dispositivos involucrados. La Fig. 9 muestra las diferentes etapas del mecanismo propuesto aplicado al caso de uso. Se han realizado las capturas de imágenes durante la presentación del contenido y se ha utilizado la herramienta Matlab para el procesado de dichas imágenes y el cálculo de las asincronías entre dispositivos según la Ec. 1.

Las Fig. 10 y 11 muestran el detalle de las imágenes tomadas por el dispositivo de entrada de vídeo del *videowall* 2x3 en funcionamiento con el contenido con fotogramas insertados (Fig. 10) y el resultado de procesado previo a la detección del número de fotograma de la imagen capturada por el dispositivo de entrada de vídeo (Fig. 11).

Para comprobar el funcionamiento correcto del mecanismo, se han tomado aproximadamente 100 capturas (imágenes), a razón de 1 imagen por segundo, de las cuales alrededor de 10 han sido descartadas al no haberse obtenido de forma clara e inequívoca el valor del número de fotograma tras el análisis de la imagen. Mayoritariamente, las medidas descartadas se correspondían con las medidas realizadas con instantes puntuales cuando aparecían los créditos de la película.

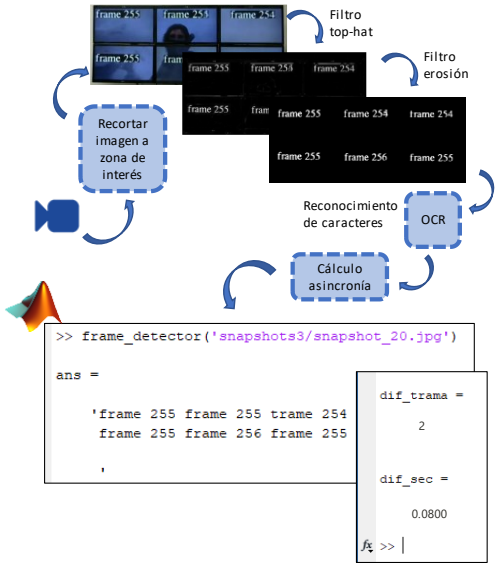

Fig. 9. Etapas del proceso de detección de los números de fotograma en el caso de uso de un sistema *videowall* 2x3.

Fig. 10. *Videowall* 2x3 reproduciendo el contenido preparado.

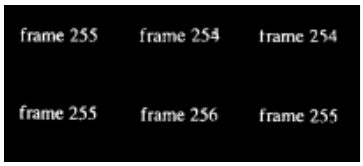

Fig. 11. Imagen procesada previa a la etapa de detección de caracteres[3].

En la Fig. 12 se observa la evolución del nivel de sincronización alcanzado y registrado por el software del propio sistema *videowall* en una de las sesiones de medida, junto a una media móvil de 10 muestras para observar más claramente su comportamiento. En la Fig. 13, se muestra la evolución de la máxima diferencia en números de fotograma evaluada durante una de las sesiones con el mecanismo propuesto en este trabajo. En particular, en la Fig. 12 puede verse cómo los valores de asincronía registrada se mantienen por debajo de los 80ms, pero se registran una serie de fluctuaciones que provocan que el valor oscile entre los -20ms y los 60ms (aproximadamente). Cabe resaltar que, para el contenido utilizado, una asincronía con valor absoluto entre]0, 40[ms puede implicar que el contenido se presente con hasta 1 fotograma de diferencia y entre [40, 80[ms con hasta 2 fotogramas de diferencia entre los dispositivos (pantallas del *videowall*) involucrados. Esto también puede comprobarse en la Fig. 13, donde se muestran las asincronías detectadas por el mecanismo propuesto, llegando a detectar un máximo de dos fotogramas de diferencia entre las pantallas involucradas. Por tanto, se puede afirmar que ambas figuras muestran resultados similares y coherentes. Además, numéricamente, el error cuadrático medio (ECM) que se obtiene de la asincronía máxima registrada por el sistema propuesto ha sido de 0.3023 fotogramas^2 con un I.C. 95% de 0.0121 fotogramas^2. A partir de dicho valor, se puede obtener el valor de la asincronía media medida en unidades de tiempo, mediante la Ec. 4:

$$Asincronía_{media} = \sqrt{ECM} * \frac{1}{fps} = 0.022s \pm 0.0043s \quad (4)$$

En la Tabla III se comparan los valores de asincronía media obtenidos, tanto a partir del propio SW del sistema *videowall* como el obtenido a través del mecanismo propuesto:

TABLA III.
VALORES DE SINCRONIZACIÓN OBTENIDOS

Origen del cálculo del nivel de sincronización	Asincronía (± 95% I.C.)
Mecanismo integrado en el SW del videowall	33 (±3.9) ms
Mecanismo propuesto	22 (±4.3) ms

Por tanto, se puede concluir que, por un lado, los resultados obtenidos son coherentes con el nivel de sincronismo que se estima en tiempos de recepción, por parte de los procesos de reproducción (SW) en los dispositivos involucrados (RPi).

De hecho, el resultado numérico corrobora esta conclusión, puesto que el valor obtenido de 22ms encaja dentro de los niveles esperados, a pesar de existir una diferencia respecto a los valores obtenidos por el propio sistema *videowall* de 11ms, la cual puede considerarse despreciable.

En definitiva, utilizando el sistema propuesto en este artículo se han podido validar de forma objetiva (de forma automatizada y sin necesidad de realizar ningún test a usuarios) las estimaciones realizadas por las técnicas de sincronización implementadas en el software del sistema de *videowall* analizado.

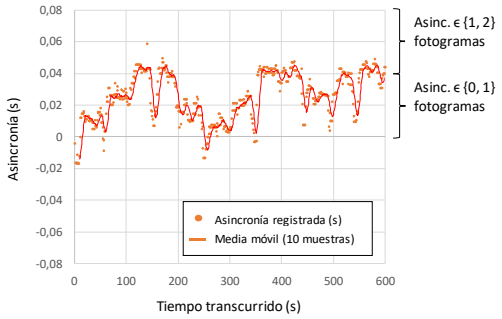

Fig. 12. Nivel de sincronización registrado por el *SW* del sistema *videowall* durante la recepción y decodificación del contenido.

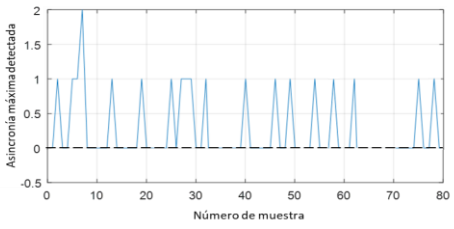

Fig. 13. Diferencia de fotogramas presentados en las del videowall durante la evaluación (dif. de 1 fotograma es equivalente a una asincronía de hasta 40ms).

[3] Se puede observar en las Figs. 9-11 que en la captura de este ejemplo, en la pantalla superior central se acabó detectando el número 255 frente al 254. Pues dicha decisión es criterio de la herramienta utilizada de Matlab, cuya mejora no se aborda en este artículo.

V. Conclusiones

En este artículo se ha presentado un mecanismo que permite medir de forma objetiva y sin estimaciones el nivel de sincronismo alcanzado en tiempos de presentación entre dos o más dispositivos que se supone que deben estar reproduciendo contenidos relacionados (o incluso los mismos) de forma sincronizada. Se trata de un sistema útil para validar técnicas de sincronización implementadas en aplicaciones SW que requieran del consumo de contenidos relacionados en una o más pantallas (escenario multi-pantalla). Debido a que, normalmente, dichas técnicas se basan en medidas llevadas a cabo durante la recepción o decodificación del contenido, y no durante la presentación del mismo, deben realizar estimaciones de la latencia asociada al proceso de reproducción hasta la visualización final del contenido en la pantalla. Con el mecanismo que se presenta en este artículo se puede medir de forma objetiva y más fidedigna el grado de sincronización adquirido en dichas aplicaciones y, de forma indirecta, validar los cálculos y estimaciones realizados en las técnicas de sincronización que se incluyen en ellas. Como demostración de la utilidad del sistema propuesto, se ha empleado para validar la técnica de sincronización empleada en un sistema de *videowall* 2x3 basado en dispositivos de bajo coste (RPis) desarrollado en el propio grupo de investigación. Se ha corroborado que los valores de sincronización medidos por el propio software de los dispositivos involucrados, a partir de las estimaciones realizadas por la solución de sincronización implementada, y el valor obtenido por el sistema propuesto son similares. Como trabajo futuro, se pretende actualizar y optimizar el mecanismo de detección de caracteres para que el número de imágenes descartadas sea el menor posible

References

[1] F. Boronat, M. Montagud, D. Marfil, and C. Luzon, "Hybrid Broadcast/Broadband TV Services and Media Synchronization: Demands, Preferences and Expectations of Spanish Consumers," *IEEE Trans. Broadcast.*, vol. 64, no. 1, 2018.

[2] M. Montagud, F. Boronat, H. Stokking, and P. Cesar, "Design, development and assessment of control schemes for IDMS in a standardized RTCP-based solution," *Comput. Networks*, vol. 70, pp. 240–259, Sep. 2014.

[3] F. Boronat, D. Marfil, M. Montagud, and J. Pastor, "HbbTV-Compliant Platform for Hybrid Media Delivery and Synchronization on Single-and Multi-Device Scenarios," *IEEE Trans. Broadcast.*, vol. 64, no. 3, 2018.

[4] D. Marfil, F. Boronat, J. López, and A. Sapena, "Mecanismo para la evaluación de la sincronización en la presentación de contenidos multi-pantalla," in *XIV Jornadas de Ingeniería Telemática (JITEL)*, 2019.

[5] J. Jansen, "VideoLat: An Extensible Tool for Multimedia Delay Measurements," in *Proceedings of the ACM International Conference on Multimedia - MM '14*, 2014, pp. 683–686.

[6] M. Hammond, J. Kramskoy, and British Broadcasting Corporation, "Measuring synchronisation timing accuracy for DVB Compainion Screen Synchronisation TVs and Companions," 2015. [Online]. Available: https://github.com/bbc/dvbcss-synctiming. [Accessed: 21-May-2019].

[7] Digital Video Broadcasting, "ETSI TS 106 286-1. Companion Screens and Streams; Part 2: Content Identification and Media Synchronization."

[8] Arduino, "Arduino Store." [Online]. Available: https://store.arduino.cc/due.

[9] M. A. Montagud Climent, F. Boronat, and P. S. César Garcia, "A customizable open-source framework for measuring and equalizing e2e delays in shared video watching," in *ACM TVX*, 2014, pp. 1–2.

[10] GStreamer, "GStreamer Framework." [Online]. Available: https://gstreamer.freedesktop.org/.

[11] GStreamer, "Videodetect Plugin." [Online]. Available: https://www.freedesktop.org/software/gstreamer-sdk/data/docs/2012.5/gst-plugins-bad-plugins-0.10/gst-plugins-bad-plugins-videodetect.html.

[12] V. K. Govindan and A. P. Shivaprasad, "Character recognition - A review," *Pattern Recognit.*, vol. 23, no. 7, pp. 671–683, Jan. 1990.

[13] A. Yousaf *et al.*, "Size invariant handwritten character recognition using single layer feedforward backpropagation neural networks," in *2019 2nd International Conference on Computing, Mathematics and Engineering Technologies, iCoMET 2019*, 2019, pp. 1–7.

[14] M. Lee, "Python-tesseract." [Online]. Available: https://pypi.org/project/pytesseract/.

[15] Accusoft, "OCRXpress." [Online]. Available: https://www.npmjs.com/package/ocr.

[16] Matlab, "Optical Character Recognition, Matlab." [Online]. Available: https://www.mathworks.com/help/vision/optical-character-recognition-ocr.html.

[17] FFmpeg, "FFmpeg." [Online]. Available: https://ffmpeg.org/. [Accessed: 22-Mar-2019].

[18] Matlab, "Top-hat filter, Matlab." [Online]. Available: https://www.freedesktop.org/software/gstreamer-sdk/data/docs/2012.5/gst-plugins-bad-plugins-0.10/gst-plugins-bad-plugins-videodetect.html.

[19] Matlab, "Erosion filter, Matlab." [Online]. Available: https://es.mathworks.com/help/images/ref/imerode.html.

[20] P. Salvador, F. Boronat, M. Montagud, and D. Marfil, "Sistema videowall de bajo coste basado en Raspberry Pi, personalizable y configurable dinámica y remotamente vía Web," in *XIII Jornadas de Ingeniería Telematica - JITEL2017*, 2017, pp. 318–325.

[21] Durian Open Movie Project, *Sintel*. https://durian.blender.org/.

Dani Marfil was born in Gandia (Spain) and studied Informatics Technical Engineering BSc degree (2011), Telecommunications BSc degree (2015) and Telecommunication Technologies, Systems and Networks MSc (2016) in Universitat Politècnica de València (UPV, Spain). He is a PhD student and an assistant researcher and developer in the Immersive Interactive Media R&D Group. His main topics of interest are communication networks, code developing and media synchronization. He is the author of one book chapter and several research and conference papers.

Fernando Boronat (M'93–SM'11) is the head of the Immersive Interactive Media R&D Group (http://iim.webs.upv.es) at the Gandia Campus of the UPV, Spain. He received the M.E. and Ph.D. degrees in telecommunication engineering from the UPV in 1994 and 2014, respectively. After working for several Spanish telecommunication companies, he moved back to the UPV in 1996. Currently, he is an Assistant Professor in its Communications Department. His main research topics of interest are immersive and interactive media systems and applications, mulsemedia and media synchronization. He is the author of two books, several book chapters, an IETF RFC and more than 100 research papers in relevant journals and conferences. He edited a book on Media Synchronization (Springer, 2018) and is involved in several IPCs of national and international refereed journals and conferences, and serves as a reviewer for highly-respected journals. He is member of IEEE (M'93–SM'11) and ACM (M'15). Contact him at fboronat@dcom.upv.es

Jair Lopez was born in La Paz, Bolivia. He received his B.Sc. degree in Telecommunication Systems, Sound and Image from the UPV, in 2019. He is currently studying the M.Sc. degree in Telecommunications Systems and Networks. His main topics of interest are communication networks and multimedia systems.

Bernardino Roig received his PhD in Mathematical Sciences from the Universitat Politècnica de Catalunya (2001) and currently, he is a professor at the Universitat Politècnica de València (2008-present). His research area focuses on computational methods applied to conservative problems, acoustic and piezoelectric processes, and also in digital image and video processing, more specifically, in image filtering and video synchronization using fuzzy metrics and numerical tools.