



UNIVERSIDAD DE CÓRDOBA



UNIVERSIDAD DE CÓRDOBA
ESCUELA POLITÉCNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

**SISTEMA INFORMÁTICO PARA EL
SEGUIMIENTO DE PUNTERO LÁSER Y
SU REPRESENTACIÓN VIRTUAL
MEDIANTE VISIÓN ARTIFICIAL**

MANUAL TÉCNICO

Autor: *Rafael Ulises Baena Herruzo*

Directores: *Juan María Palomo Romero*
Lorenzo Salas Morera

Córdoba, septiembre de 2017

Fdo: Rafael Ulises Baena Herruzo

Córdoba, septiembre de 2017

D. Juan María Palomo Romero, Colaborador Honorario del Área de Conocimiento de Proyectos de Ingeniería y adscrito al Departamento de Ingeniería Rural de la Universidad de Córdoba, y **Dr. Lorenzo Salas Morera**, Profesor Titular de Universidad del Área de Conocimiento de Proyectos de Ingeniería y adscrito al Departamento de Ingeniería Rural de la Universidad de Córdoba,

INFORMAN

Que el presente Trabajo Fin de Grado, titulado **Sistema Informático para el Seguimiento de Puntero Láser y su Representación Virtual mediante Visión Artificial**, ha sido realizado, bajo su dirección, por el alumno Rafael Ulises Baena Herruzo, reuniendo, a su juicio, las condiciones exigidas en este tipo de trabajos.

D. Juan María Palomo Romero

Dr. Lorenzo Salas Morera

Córdoba, septiembre de 2017

Índice general

Índice de figuras.....	V
Índice de tablas.....	VII
Capítulo 1. Introducción.....	1
Capítulo 2. Definición del problema.....	3
2.1 Definición del problema real.....	3
2.2 Definición del problema técnico.....	4
2.2.1 Funcionamiento.....	4
2.2.2 Entorno.....	6
2.2.3 Vida esperada.....	7
2.2.4 Ciclo de mantenimiento.....	8
2.2.5 Competencia.....	8
2.2.6 Aspecto externo.....	9
2.2.7 Estandarización.....	10
2.2.8 Calidad y fiabilidad.....	10
2.2.9 Programa de tareas.....	11
2.2.9.1 Investigación del problema.....	11
2.2.9.2 Búsqueda de tecnologías que faciliten el trabajo.....	11
2.2.9.3 Análisis de requisitos.....	11
2.2.9.4 Diseño.....	11
2.2.9.5 Desarrollo e implementación del sistema.....	12
2.2.9.6 Pruebas del sistema.....	12
2.2.9.7 Documentación.....	12
2.2.10 Pruebas.....	13
2.2.11 Seguridad.....	13
Capítulo 3. Objetivos.....	17
Capítulo 4. Antecedentes.....	19

II Índice

4.1 Tratamiento digital de imágenes.....	19
4.1.1 Tipos de imagen digital.....	20
4.1.2 El color.....	21
4.1.3 El espacio de color RGB.....	22
4.1.4 Técnicas de pre-procesado de la imagen.....	23
4.1.4.1 Histograma: Thresholding:.....	24
4.1.4.2 Umbralización:.....	25
4.1.5 Técnicas de post-procesado de la imagen.....	25
4.1.5.1 Procesado morfológico:.....	25
4.1.5.2 Dilatación y erosión.....	26
4.2 Detección de movimiento en imágenes.....	28
Capítulo 5. Restricciones.....	31
5.1 Factores dato.....	31
5.2 Factores estratégicos.....	32
Capítulo 6. Recursos.....	35
6.1 Recursos humanos.....	35
6.2 Recursos hardware.....	35
6.3 Recursos software.....	36
Capítulo 7. Especificación de requisitos.....	37
7.1 Requisitos de información.....	37
7.2 Requisitos funcionales.....	38
7.3 Requisitos no funcionales.....	40
Capítulo 8. Análisis del sistema.....	43
8.1 Análisis de los casos de uso.....	43
8.1.1 Caso de uso 0: Sistema de captura y representación de puntero láser.....	44
8.1.2 Caso de uso 1: Gestionar captura.....	45
8.1.1 Caso de uso 2: Gestionar montaje.....	50
8.1.2 Caso de uso 3: Gestionar ajustes.....	53
8.2 Diagramas de secuencia.....	56
8.2.1 Diagrama de secuencia 1: Gestionar captura.....	56

8.2.2 Diagrama de secuencia 2: Gestionar montaje.....	58
Capítulo 9. Especificación de la interfaz.....	61
9.1 Introducción.....	61
9.2 Característica de la interfaz.....	62
9.2.1 Pestaña captura.....	62
9.2.2 Pestaña montaje.....	64
9.2.3 Pestaña ajustes.....	65
Capítulo 10. Diseño del sistema.....	67
10.1 Diseño de datos.....	67
10.1.1 Fichero de captura.....	67
10.1.2 Fichero de audio.....	68
10.1.3 Fichero de vídeo de la presentación.....	69
10.1.4 Fichero de vídeo procesado.....	69
10.1.5 Fichero de vídeo final.....	70
10.2 Diseño procedimental.....	70
10.2.1 Capa InterfazUsuario.....	71
10.2.1.1 Clase VentanaPrincipal.....	71
10.2.2 Capa lógica.....	77
10.2.2.1 Clase tracker.....	77
10.2.2.2 Clase Grabacion.....	83
10.2.2.3 Clase LogicClass.....	86
10.2.2.4 Clase Clock.....	87
10.3 Diseño de la Interfaz de Usuario.....	89
10.3.1 Pestaña captura.....	90
10.3.2 Pestaña montaje.....	92
10.3.3 Pestaña ajustes.....	94
10.3.4 Pestaña información.....	96
Capítulo 11. Pruebas.....	99
11.1 Pruebas de Casos de Uso.....	99
11.1.1 Caso de prueba. Seleccionar medios de captura de vídeo.....	99
11.1.2 Caso de prueba. Previsualizar medios de captura de vídeo.....	100
11.1.3 Caso de prueba. Seleccionar dispositivo de captura de audio.....	101
11.1.4 Caso de prueba. Iniciar captura.....	101

IV Índice

11.1.5 Caso de prueba. Seleccionar superficie de proyección.....	102
11.1.6 Caso de prueba. Seleccionar ruta de destino.....	102
11.1.7 Caso de prueba. Iniciar montaje.....	103
11.1.8 Caso de prueba. Seleccionar relación de aspecto.....	103
11.1.9 Caso de prueba. Usar pantalla extendida.....	104
11.1.10..... Caso de prueba. Montaje offline - guardar captura	105
11.1.11..... Caso de prueba. Montaje offline - cargar captura	105
11.2 Pruebas de Escenarios de la Aplicación.....	106
11.2.1 Caso de prueba. Seleccionar medios de captura de vídeo.....	106
11.2.2 Caso de prueba. Previsualizar medios de captura de vídeo.....	107
11.2.3 Caso de prueba. Seleccionar dispositivo de captura de audio.....	107
11.2.4 Caso de prueba. Seleccionar superficie de proyección.....	108
11.2.5 Caso de prueba. Seleccionar ruta de destino.....	108
11.2.6 Caso de prueba. Iniciar montaje.....	108
11.2.7 Caso de prueba. Montaje offline – guardar captura.....	109
11.2.8 Caso de prueba. Montaje offline – cargar captura.....	109
Capítulo 12. Conclusiones.....	111
12.1 Conclusiones sobre los objetivos planteados.....	111
12.2 Conclusiones de la fase de pruebas.....	112
12.3 Futuras mejoras.....	113
Bibliografía.....	115

Índice de figuras

Figura 4.1: Comparación de las regiones curvas generadas mediante una imagen vectorial y un mapa de bits.....	21
Figura 4.2: Espectro visible de la luz.....	22
Figura 4.3: Colores RGB.....	23
Figura 4.4: Histograma imagen en escala de grises.....	24
Figura 4.5: Umbralización utilizando diferentes valores de umbral.....	25
Figura 4.6: Elementos estructurantes.....	26
Figura 4.7: Ejemplo erosión.....	27
Figura 4.8: Ejemplo de dilatación.....	27
Figura 8.1: Diagrama de caso de uso 0: Sistema de captura y representación de puntero láser.....	44
Figura 8.2: Diagrama de caso de uso 1: Gestionar captura.....	45
Figura 8.3: Diagrama de caso de uso 2: Gestionar montaje.....	50
Figura 8.4: Diagrama de caso de uso 3: Gestionar ajustes.....	53
Figura 8.5: Diagrama de secuencia 1: Gestionar captura.....	55
Figura 8.6: Diagrama de secuencia 2: Gestionar montaje.....	59
Figura 9.1: Estructura de pestañas de la ventana principal de la interfaz.....	62
Figura 9.2: Estructura de la pestaña captura de la interfaz.....	63
Figura 9.3: Estructura de la pestaña montaje de la interfaz.....	64
Figura 9.4: Estructura de la pestaña ajustes de la interfaz.....	65
Figura 10.1: Componente correspondiente con la interfaz de usuario.....	71
Figura 10.2: Clase VentanaPrincipal.....	76
Figura 10.3: Componente correspondiente con la capa lógica.....	77
Figura 10.4: Clase Tracker.....	82
Figura 10.5: Clase Grabacion.....	86
Figura 10.6: Clase LogicClass.....	87
Figura 10.7: Clase Clock.....	88

VI Índice

Figura 10.8: Diagrama de clases.....	89
Figura 10.9: Pestaña captura.....	90
Figura 10.10: Ventana buscar archivo.....	91
Figura 10.11: Pestaña montaje.....	93
Figura 10.12: Pestaña ajustes.....	96
Figura 10.13: Pestaña información.....	97

Índice de tablas

Tabla 2.1: Fases del proyecto y distribución temporal.....	15
Tabla 8.1: Plantilla de casos de uso.....	43
Tabla 8.2: Caso de uso 1.1: seleccionar medios de captura de vídeo.....	46
Tabla 8.3: Caso de uso 1.2: previsualizar medios de captura de vídeo.....	47
Tabla 8.4: Caso de uso 1.3: seleccionar dispositivo de captura de audio.....	48
Tabla 8.5: Caso de uso 1.4: iniciar captura.....	49
Tabla 8.6: Caso de uso 1.5: seleccionar superficie de proyección.....	49
Tabla 8.7: Caso de uso 2.1: seleccionar ruta de destino.....	51
Tabla 8.8: Caso de uso 2.2: iniciar montaje.....	52
Tabla 8.9: Caso de uso 3.1: seleccionar relación de aspecto.....	54
Tabla 8.10: Caso de uso 3.2: usar pantalla extendida.....	54
Tabla 8.11: Caso de uso 3.3: montaje offline – guardar captura.....	55
Tabla 8.12: Caso de uso 3.4: montaje offline – cargar captura.....	56
Tabla 11.1: Caso de prueba: Seleccionar medios de captura de vídeo.....	100
Tabla 11.2: Caso de prueba: Previsualizar medios de captura de vídeo.....	100
Tabla 11.3: Caso de prueba: Seleccionar dispositivo de captura de audio.....	101
Tabla 11.4: Caso de prueba: Iniciar captura.....	101
Tabla 11.5: Caso de prueba: Seleccionar superficie de proyección.....	102
Tabla 11.6: Caso de prueba: Seleccionar ruta de destino.....	102
Tabla 11.7: Caso de prueba: Iniciar montaje.....	103
Tabla 11.8: Caso de prueba: Seleccionar relación de aspecto.....	104
Tabla 11.9: Caso de prueba: Usar pantalla extendida.....	104
Tabla 11.10: Caso de prueba: Montaje offline – guardar captura.....	105
Tabla 11.11: Caso de prueba: Montaje offline – cargar captura.....	106
Tabla 11.12: Pruebas de Escenarios de la Aplicación: Seleccionar medios de captura de vídeo.....	107

Tabla 11.13: Pruebas de Escenarios de la Aplicación: Previsualizar medios de captura de vídeo.....	107
Tabla 11.14: Pruebas de Escenarios de la Aplicación: Seleccionar dispositivo de captura de audio.....	107
Tabla 11.15: Pruebas de Escenarios de la Aplicación: Seleccionar superficie de proyección.....	108
Tabla 11.16: Pruebas de Escenarios de la Aplicación: Seleccionar ruta de destino.....	108
Tabla 11.17: Pruebas de Escenarios de la Aplicación: Iniciar montaje.....	108
Tabla 11.18: Pruebas de Escenarios de la Aplicación: Montaje offline – guardar captura.....	109
Tabla 11.19: Pruebas de Escenarios de la Aplicación: Montaje offline – cargar captura.....	109

Capítulo 1. Introducción

Una presentación digital es un producto multimedia compuesto por una serie de imágenes que pueden ir acompañadas no sólo de texto sino también de gráficos y de interactividad, y que se muestran en forma de diapositivas. Este recurso, perteneciente a la era de la informática, permite la exposición de un tema de una forma llamativa, rápida y profesional.

Las presentaciones tienen su origen en el clásico uso de diapositivas fotográficas en exposiciones y presentaciones de negocios, académicas y científicas. Normalmente un expositor explicaba o narraba la información de la presentación mientras esta era apoyada visualmente con transparencias ampliadas desde un proyector de diapositivas de 35 mm. Posteriormente esta técnica fue avanzando, incorporando música de narraciones con explicaciones grabadas con *casete* y usando de manera combinada dos o más proyectores para conseguir algunos efectos de transición o animaciones.

A mediados de la década de los ochenta se crea *Presenter* el primer programa de ordenador capaz de producir presentaciones gráficas. Posteriormente en 1987 este fue vendido a Microsoft Corporation donde evolucionó hasta lo que se conoce hoy día como *Power Point*.

En la actualidad muchas de estas presentaciones se transmiten a través de distintas plataformas de Internet, para así poder llegar a un mayor número de personas. Generalmente, las presentaciones físicas son grabadas y retransmitidas a través de Internet, en plataformas como *YouTube*, mediante cámaras web, las cuales suelen disponer de una resolución insuficiente que permita una visualización óptima.

Aunque en entornos más profesionales se utilizan cámaras de mayor calidad, así como un número considerable de recursos humanos, la calidad de la grabación nunca llegará a ser tan nítida como la de la propia presentación en digital. Con el uso de montaje de vídeo (ya sea en tiempo real o en posproducción) se puede resolver este problema, pero seguirá existiendo un asunto no trivial, el seguimiento e integración del puntero láser, el dispositivo más usado en la actualidad a la hora de realizar cualquier tipo de presentación, herramienta vital como apoyo del mensaje oral, así como para guiar la atención de los espectadores durante la presentación.

Se hace por ello necesario el desarrollo de una aplicación que permita la captación automática y en tiempo real, de un puntero láser (físico) y su transformación fidedigna e integración en una grabación digital de la presentación. Todo esto, con la ayuda de la visión artificial.

Capítulo 2. Definición del problema

En este punto se tratará de mostrar detalladamente el problema al que se pretende dar solución con la realización del proyecto. Para definir el problema, se distingue entre el problema real, el cuál trata sobre la propia visión ofrecida por el usuario, y el problema técnico, que define el problema desde el punto de vista de la ingeniería.

2.1 Definición del problema real

En el ámbito de las presentaciones electrónicas, se encuentran carencias a la hora de transmitir estas a terceras personas, por lo general, a través de plataformas de vídeo en línea. Esto ocurre sobremanera al realizar la grabación de la presentación con los instrumentales típicos, ya sean cámaras de vídeo y sistemas de audio. En este sentido la cantidad de recursos humanos y hardware requeridos puede llegar a ser desmesurada, debido a que hace falta personal para poder hacer el montaje en tiempo real y la calidad se va a ver limitada por el hardware de grabación usado.

Muchas veces se opta por la opción más sencilla, grabar solamente las diapositivas originales desde el ordenador que las proyecta, para así obtener la máxima calidad de vídeo. El problema de este sistema es que el espectador pierde integración con la presentación, al no disponer de la ayuda visual del puntero láser del expositor e incluso la propia imagen de este.

Con la realización de este proyecto se pretende dar solución al problema que existe a la hora de grabar presentaciones, principalmente la baja calidad que estas presentan (debido generalmente a los dispositivos de captación empleados), conllevando a una baja integración por parte de los espectadores en dichas presentaciones. Para ello se le va a otorgar al orador (el usuario del sistema informático) un programa que satisfaga la necesidad de captar el puntero láser físico y su conversión en uno virtual, preservando la máxima calidad de las diapositivas y añadiendo la posibilidad de grabar el audio del propio orador.

2.2 Definición del problema técnico

Para identificar el problema técnico se utilizará una técnica de ingeniería denominada PDS (*Product Design Specification*), una metodología que permite realizar un análisis de los principales condicionantes técnicos del problema, dando respuesta a la formulación de una serie de preguntas que se detallan a continuación.

2.2.1 Funcionamiento

En este apartado se van a analizar las principales prestaciones que debe reunir el sistema informático que se pretende implementar desde un punto de vista técnico:

- El sistema debe permitir cargar un archivo que contenga el un presentación ya grabada, en caso de que el usuario no quiera realizar la captación en directo.
- El sistema debe permitir seleccionar una cámara, del conjunto total de dispositivos de captación de imágenes conectados al ordenador.
- El sistema debe poder previsualizar el contenido de cada una de las cámaras conectadas, para que así no haya cabida a equivocaciones.

- El sistema deberá permitir seleccionar un dispositivo de captación de sonido, del conjunto total de micrófonos conectados al ordenador.
- El sistema debe permitir al usuario visualizar el conjunto de dispositivos de entrada de sonido de forma gráfica y sencilla.
- El sistema debe ser capaz de crear ficheros “.cap” para guardar las coordenadas del puntero.
- El sistema debe ser capaz de crear ficheros “.mp4” que guardarán el resultado final del montaje.
- El sistema debe ser capaz de crear ficheros con extensión “.mp3” que almacenarán el audio generado por el usuario.
- La aplicación debe permitir la ejecución de la captura de vídeo de la presentación, la captura de audio, la obtención de las coordenadas del puntero láser y el montaje de la presentación, creando un fichero “.mp4” que contenga las diapositivas, el puntero representado digitalmente y el audio.
- El sistema debe poder realizar un seguimiento del puntero láser del mundo real.
- El sistema debe permitir al usuario la selección de la superficie de proyección¹ realizando una extracción con el cursor del ordenador.
- La aplicación deberá dar un feedback al usuario cuando esté realizando el montaje, mediante elementos que muestren el progreso de este.
- El sistema debe permitir al usuario la selección de la relación de aspecto que tienen las diapositivas que se están mostrando en la presentación.
- La aplicación debe permitir al usuario la utilización de la pantalla de forma extendida, para ello facilitará la introducción de la resolución de la pantalla principal.

¹ La superficie de proyección corresponde con el área física donde se muestran las diapositivas.

- El sistema debe mostrar una interfaz lo más sencilla e intuitiva posible, facilitando así la utilización de la aplicación por parte del usuario.

2.2.2 Entorno

El entorno se define como el conjunto de aspectos o propiedades que rodean al problema y que, aún presentándose de manera externa al mismo, pueden influir en el planteamiento de una solución, puesto que pueden afectar al sistema software desarrollado para tal fin.

En el análisis del entorno de la aplicación software que se pretende desarrollar se tendrán en cuenta los siguientes puntos de vista: entorno de programación, entorno software, entorno hardware y entorno de usuario.

- Entorno de programación: la presente aplicación será desarrollada en el lenguaje de programación C++, haciendo uso del editor de código fuente Atom [2], desarrollado por *Github* y del IDE (*Entorno de Desarrollo Integrado*) *Qt Creator* para el desarrollo de la interfaz gráfica. También se va a utilizar la biblioteca libre de visión artificial *OpenCV* [4].
- Entorno software: para el correcto funcionamiento de la aplicación serán necesarios los siguientes componentes software:
 - Será necesario disponer de la biblioteca libre de visión artificial *OpenCV* y de las librerías *Libav* [3] y *Qt* [5], en el apartado *recursos* se detallará más esta información.
 - La naturaleza de C++ hace que pierda la componente de multiplataforma, debiendo algunas partes del código, desarrollarse de manera específica para cada sistema operativo.
 - El sistema informático será desarrollado para ser ejecutado en sistemas *GNU/Linux*.
- Entorno hardware: también conocido como el entorno físico o de trabajo, hace referencia a las características del sistema informático en el que se ejecutará la aplicación, así como el ambiente que lo rodea. Es

software a desarrollar se instalará en un ordenador personal, existiendo en principio unos requisitos moderados con respecto a los recursos de memoria, almacenamiento y velocidad de los que deberá disponer; será necesario que el rendimiento del mismo resulte adecuado para poder hacer uso de la aplicación de manera correcta. En cuanto al entorno en el que se encontrará dicho ordenador personal, se considerará que está en condiciones propicias, cuando haya una iluminación adecuada para la detección del puntero láser.

- Entorno de usuario: el sistema informático a desarrollar deberá ser lo más intuitivo posible para permitir a el usuario una correcta elaboración del vídeo final. Los usuarios finales deben tener mínimos conocimiento informáticos, tales como conocer la resolución de su pantalla y distinguir entre relaciones de aspecto, conocimientos que son necesarios siempre que se quiera hacer un uso del programa más completo.

2.2.3 Vida esperada

La vida esperada de un producto software puede definirse como el tiempo estimado durante el cual puede realizarse una aplicación útil del mismo. Esta estimación es compleja de realizar ya que influyen diversos factores en la misma. Aun así, se va a realizar un análisis de las condiciones de mayor relevancia para su estimación:

- El lenguaje de programación C++, en el que se desarrollará el sistema software, se encuentra en continua evolución. Actualmente la versión actual es la *C++14* y se estima que a finales de 2017, será *C++17*, por lo que sería de utilidad realizar actualizaciones de manera periódica, que permitan un mayor rendimiento del sistema.
- La biblioteca de visión artificial *OpenCV*, suele actualizarse con menor frecuencia, actualmente se encuentra en la versión *3.3.0*. Aun así,

cuenta con una rama de desarrollo paralela, que podría traer nuevas prestaciones que faciliten el funcionamiento de la aplicación.

- La librería *libav*, utilizada para realizar la grabación de las diapositivas, el audio y su posterior conversión a vídeo, podría verse sometida a nuevas actualizaciones por parte del *Libav team*, pudiendo mejorar la funcionalidad de la versión actual.

2.2.4 Ciclo de mantenimiento

El ciclo de mantenimiento trata sobre el conjunto de modificaciones que una aplicación puede soportar frente a las diversas circunstancias que puedan surgir, o nuevas exigencias procedentes por parte del sistema o del usuario final.

El mantenimiento se debe llevar a cabo por programadores informáticos, atendiendo a los siguientes tipos de necesidades:

- Perfeccionamiento: conjunto de funciones que mejoren la funcionalidad del programa. Aumentando, entre otras, la eficiencia, la facilidad de mantenimiento para futuros cambios, etc.
- Adaptación: conjunto de actividades realizadas para adaptar la aplicación al entorno tecnológico del momento, pudiendo cambiar la forma de guardar la información de la aplicación (capturas del puntero, archivos de audio y video), los métodos para captar el puntero láser, etc.
- Corrección: conjunto de actividades necesarias para llevar a cabo las correcciones pertinentes para solucionar errores no detectados en la aplicación.

2.2.5 Competencia

Actualmente, existen muchas aplicaciones que sirven para realizar un seguimiento de objetos en tiempo real, aunque hay algunas aproximaciones software a este sistema informático, ninguna de ellas supone una competencia

real, ya que este sistema software tiene unos aspectos y un enfoque bastante diferenciados del resto.

Estos programas, son en su mayoría asistentes para la realización de la presentación, sin ahondar en funcionalidades como las que aquí se exponen (captación y representación del puntero láser) y sin usar técnicas de visión artificial.

2.2.6 Aspecto externo

La apariencia externa de una aplicación hace referencia no solamente al aspecto visual que tiene el usuario de la misma durante su ejecución, sino que también es necesario considerar la presentación física del sistema, los mecanismos de instalación proporcionados junto al mismo y los manuales que pueden acompañarlo.

Se consideran los siguientes aspectos:

- Interfaz de usuario: la interfaz de usuario de la aplicación, debe facilitar el uso por parte del usuario, siendo necesario que ésta sea lo más intuitiva posible. Para ello se hace uso de *Qt* el cual es un framework multiplataforma orientado a objetos, utilizado para generar interfaces gráficas de usuario. Por otra parte se va a hacer uso de las propias herramientas que *OpenCV* proporciona para el visionado de la detección del puntero láser.
- Formato de almacenamiento: el dispositivo de almacenamiento en el cual se va a entregar es un CD-ROM, debido a sus cualidades de portabilidad, bajo coste, seguridad, resistencia, uso generalizado entre los usuarios, etc. En el CD-ROM irán incluidos tanto los archivos correspondientes con el manual del software, como los necesarios para la ejecución del programa.
- Documentación: la documentación de la aplicación (la cuál irá incluida en el CD-ROM) estará formada por el Manual Técnico, en el que se

engloba toda la información del proceso de análisis, diseño, implementación y prueba del sistema; el Manual de Código, en el que se incluirá información detallada de la implementación del software final; y el Manual de Usuario en el que se detallará de una forma sencilla el acceso a la aplicación y el uso de todos los componentes de la misma.

2.2.7 Estandarización

Para el desarrollo de la aplicación se va a utilizar el lenguaje de programación *C++* haciendo uso de la biblioteca *OpenCV*, las librerías *libav*, *libX11* (para la detección de las propiedades de la pantalla) y *librt* (para manejar el tiempo). Todas estas librerías y herramientas están bastante extendidas y no representan problema alguno para la estandarización.

Cuando se hace uso del lenguaje de programación *C++* [1], es recomendable seguir unas normas (estándar) respecto al estilo de codificación, tal y como ocurre en otros lenguajes de programación.

2.2.8 Calidad y fiabilidad

En cuanto a la calidad y fiabilidad de una aplicación, son factores muy importantes, ya que el usuario debe tener unas garantías del funcionamiento del software que está usando.

El punto de mayor riesgo que puede tener el programa, es a la hora grabar el audio y el vídeo de la presentación con la librería *libav* ya que se realiza una ejecución externa de dicho programa y no se podrán comprobar los posibles errores que puedan surgir de manera directa. Para solucionar esto, el sistema informático hará uso de otras técnicas, como la comprobación de que se han creado correctamente los archivos de salida, para comprobar la consistencia de este, y en consecuencia su calidad y fiabilidad. Destacar que, a priori, los únicos errores que debería cometer el sistema serán errores producidos por el uso incorrecto por parte del usuario. También se tendrán en cuenta aspectos referentes a la calidad del diseño y la estética del programa, como su funcionamiento.

2.2.9 Programa de tareas

El programa de tareas se define como el conjunto de actividades y etapas que constituyen el proceso de desarrollo de una aplicación. A continuación se va a proceder a describir las distintas fases en las que se va a organizar el desarrollo del proyecto.

2.2.9.1 Investigación del problema

En esta primera fase, se definirá el dominio del problema, incluyéndose una descripción de la situación actual, restricciones del problema y metas que se lograrán. En particular, se hará un estudio detenido del sistema PowerPoint y los diferentes sistemas de seguimiento de punteros láser y grabación de vídeo.

2.2.9.2 Búsqueda de tecnologías que faciliten el trabajo

En esta segunda fase, se realizará un análisis de las principales herramientas de visión artificial para la detección de movimiento y grabación de audio y vídeo.

2.2.9.3 Análisis de requisitos

En esta tercera fase, deben identificarse aquí las funciones que deberá realizar el software, el rendimiento esperado y las interfaces necesarias.

2.2.9.4 Diseño

En esta fase, se traducirán las características formales generales de la aplicación, tales como la estructura de los datos, la arquitectura del software, la caracterización del interfaz y los procedimientos, recogidos de la etapa anterior.

2.2.9.5 Desarrollo e implementación del sistema

En esta fase, se traducirán todas las especificaciones de la fase anterior a un lenguaje entendible por el hardware en donde se va a instalar la aplicación. Principalmente los esfuerzos se centrarán en la implementación de un algoritmo para la detección del puntero láser.

2.2.9.6 Pruebas del sistema

En esta fase, se comprobará que el funcionamiento de la aplicación sea correcto, de tal manera que todas las entradas posibles generen las salidas deseadas.

2.2.9.7 Documentación

De cada una de estas fases se elaborará una documentación en donde quedará constancia de todo el proceso realizado y los pasos seguidos. Esencialmente se elaborarán los tres documentos que corresponden a un proyecto de ingeniería informática:

- **Memoria:** es un documento que contiene la descripción del problema y las soluciones que se han adoptado para llevar a cabo el proyecto.
- **Manual de usuario:** en este documento se indicarán los pasos para poder instalar y utilizar el programa, además de una guía para que cualquier persona, no experta, aprenda a usarla de la forma más sencilla y concisa. Para su mayor comprensión, se va a acompañar con ejemplos e ilustraciones.
- **Manual de código:** en este documento se mostrará el código fuente de la aplicación, con la finalidad de que una persona ajena al proyecto pueda identificar con facilidad el significado de las variables, y el funcionamiento de las distintas clases y métodos. Para su mayor comprensión se harán comentarios en el código.

La temática y el proyecto a realizar se ha escogido teniendo en cuenta que la duración del mismo debe ocupar al alumno 300 horas de trabajo, las

asigna el plan de estudios. Son equivalentes a los 12 créditos ECTS que debe cubrir. El reparto se realizará siguiendo la distribución en fases descrita anteriormente. Dado que estas fases no son secuenciales, sino que pueden ser paralelas y retroalimentarse, se procederá a hacer un reparto en horas de trabajo.

Esta planificación descrita es a priori, por lo que podrían ocurrir variaciones en cuanto a la duración de esta, por otro lado, con el estudio realizado se ha llegado a la conclusión de que son las estimaciones más probables.

En la *Tabla 2.1* se pueden observar las principales fases de desarrollo y distribución temporal del proyecto.

2.2.10 Pruebas

En la fase de pruebas se verificará que se cumplan las especificaciones que se planteen en el apartado de requisitos para así eliminar los posibles errores que se hayan cometido durante el desarrollo de la aplicación.

Se probará de manera exhaustiva el algoritmo de detección de movimiento, haciendo uso de diversas pruebas, entre ellas probar diferentes condiciones de iluminación en la sala y pruebas que midan la efectividad del algoritmo al colocar la cámara a diferentes distancias de la superficie de proyección.

Todas las pruebas realizadas serán debidamente documentadas en el apartado correspondiente de *pruebas*.

2.2.11 Seguridad

La seguridad que tendrá el sistema informático que se pretende desarrollar, deberá consistir en garantizar que en la ejecución no se realice ninguna actividad incorrecta ajena a su propia funcionalidad. Además, será necesario garantizar la seguridad de los dispositivos de almacenamiento, ya que se van a realizar entradas y salidas de datos en estos.

Por otra parte, no hará falta establecer medidas de seguridad referentes a proteger la confidencialidad de los datos, ya que el usuario no deberá proporcionar ningún tipo de información personal, careciendo estos de privacidad. Tampoco se deberá realizar ninguna acción de protección contra copia.

Tareas	Tiempo (h)
Investigación del problema	35
+ PowerPoint	5
+ Seguimiento de punteros láser	10
+ Investigar nuevas técnicas y posibilidades	20
Búsqueda de tecnologías que faciliten el trabajo	30
+ OpenCV	10
+ Algoritmos de detección de bordes	5
+ Familiarización de la programación con <i>C++</i>	15
Análisis de requisitos	10
+ Funciones a realizar por el software	5
+ Análisis de interfaces necesarias	5
Diseño	65
+ Diseño de las estructuras de datos	40
+ Diseño de los modelos computacionales	25
Desarrollo e implementación del sistema	60
+ Programación de las estructuras de datos	35
+ Programación de los modelos computacionales	25
Pruebas del sistema	50
+ Diseño de las pruebas	10
+ Implementación de las pruebas	15
+ Análisis de los resultados	25
Documentación	50

+ Manual técnico	35
+ Manual de usuario	10
+ Manual de código	5
Total	300

Tabla 2.1: Fases del proyecto y distribución temporal

Capítulo 3. Objetivos

El objetivo principal de este proyecto será el desarrollo e implementación de un sistema informático que permita la grabación de una exposición pública (conferencias, congresos, docencia, etc) con ayuda del proyector, representando virtualmente el seguimiento del puntero láser físico utilizado como apoyo de la presentación. Para ello, el sistema será capaz de reconocer objetos en movimiento (en este caso un puntero láser físico).

Será necesario cumplir con los siguientes objetivos:

- Desarrollar un algoritmo que sea capaz de capturar el contenido de una videocámara y de localizar las coordenadas de un puntero láser situado en dichas capturas.
- Implementar un algoritmo que realice la grabación del audio y las dispositivas de una presentación.
- Desarrollar un método para procesar las coordenadas obtenidas y aplicárselas correctamente a las diapositivas capturadas desde el ordenador, de tal forma que represente de forma fidedigna el lugar al que está apuntando el conferencista (orador en la presentación).
- Diseñar una interfaz de usuario fácil e intuitiva.
- Implementar un método capaz de generar un fichero con el montaje final, en un archivo de vídeo.

- Crear la posibilidad de realizar un montaje *offline* del vídeo final, de esta forma el usuario no tendrá que realizar el montaje en directo (evitando así la posible demora de tiempo que esto pudiese conllevar).
- Permitir a el usuario la elección de manera manual o automática de la relación de aspecto en la que se están proyectando las diapositivas.
- Permitir la creación de una grabación final realizada con los *códecs*² necesarios para poder compartirla a través de plataformas web de compartición de vídeo como *Youtube*, para que cualquier usuario pueda reproducirla sin necesidad de software adicional.
- Adaptar el sistema para que pueda ser accesible y usable para todo tipo de usuarios.

² Un códec es un programa o dispositivo hardware capaz de codificar o decodificar una señal o flujo de datos digitales. Su uso está muy extendido para la codificación de señales de audio y vídeo dentro de un formato contenedor. [7]

Capítulo 4. Antecedentes

Los antecedentes del presente proyecto se pueden clasificar en dos categorías bien diferenciadas:

- El tratamiento digital de imágenes.
- La detección de movimiento en imágenes.

4.1 Tratamiento digital de imágenes

Ha habido un gran avance tecnológico en la adquisición y el tratado de imágenes digitales en los últimos años, esto sumado a su profunda y cada vez más estrecha relación con los ordenadores, permite la realización de estudios de las características de la imagen cada vez más detallados y especificados. Hoy en día, se puede encontrar en casi cualquier hogar, un ordenador, una cámara o un escáner; y aunque los equipos son cada vez más complejos y sofisticados, el manejo a nivel de usuario de estos, se ha ido adaptando a una ejecución casi automática y simplificada. Esto se debe, a que entender la tecnología detrás del tratamiento de imágenes digitales, se hace accesible mediante un alto conocimiento de sus fundamentos básicos.

El tratamiento digital de imágenes estudia los procesos y técnicas necesarios para manipular la imagen de forma que el resultado, permita descubrir o resaltar cierta información que dicha imagen contenga.

En este apartado se van a poner en antecedentes algunos conceptos básicos y técnicas del tratamiento digital de imágenes, que servirán para comprender el sistema implementado en este proyecto.

4.1.1 Tipos de imagen digital

A la hora de clasificar imágenes, se pueden estudiar de múltiples formas y criterios. Este apartado se centra en las imágenes descritas en un ordenador. En cualquier procesamiento digital, la adquisición de la imagen a través de un sistema compuesto por sensores ópticos, es la primera fase. Existen dos grupos en los que se pueden clasificar las imágenes digitales: imágenes vectoriales e imágenes en mapas de bits.

Una imagen vectorial, está compuesta por un conjunto de contornos y rellenos que a su vez están descritos mediante formulación matemática. La principal ventaja que tienen este tipo de imágenes, es que pueden ser escalables sin perder ni información ni calidad en ellas. En contra de esto, conforme más compleja es una imagen, mas lo son las ecuaciones que la definen y mayor es el tiempo de cómputo necesario y el peso en la memoria del sistema. Debido a esto, el principal cometido de este tipo de imágenes es la representación de gráficas, caracteres y estructuras geométricas, ya que contienen elementos claramente clasificables y diferenciables.

Las imágenes de mapas de bits, son descritas mediante píxeles, pequeños cuadrados de igual tamaño, los cuales recorren toda la imagen. Cada uno de estos píxeles guardan la información correspondiente con el color que tiene la imagen en ese punto concreto, dando lugar a una malla representativa de toda la imagen. Como se puede observar en la *figura 4.1* se puede ver la diferencia entre estos dos tipos de formato de imagen. Viendo como en las regiones curvas de un mapa de bits, se generan estructuras dentadas.

Las imágenes de mapas de bits, tienen una gran pérdida en la nitidez cuando se realizan escalados en esta, pero en contraposición ganan una gran diversidad en la gama de color y en el tono. Debido a que el sistema informático que se pretende implementar en este proyecto, pretende reconocer

un objeto del mundo real mediante la captura de imágenes por *Webcam*, en un entorno con una amplia gama de colores y saltos y debido a que el escalado no es importante, se utilizarán imágenes en mapas de bits.



Figura 4.1: Comparación de las regiones curvas generadas mediante una imagen vectorial y un mapa de bits

4.1.2 El color

El color es una interpretación subjetiva e individual de las ondas electromagnéticas que refleja un cuerpo expuesto a una fuente de iluminación. Del conjunto de longitudes de onda que se pueden encontrar dentro del espectro electromagnético de la luz solar, el ojo humano es capaz de detectar solo el espectro visible, compuesto por aquellas ondas que se encuentran comprendidas entre los 380 y 780 nanómetros. Dentro de este espectro, se pueden clasificar en franjas de color los 6 colores espectrales y en los límites, el resto de tonalidades.

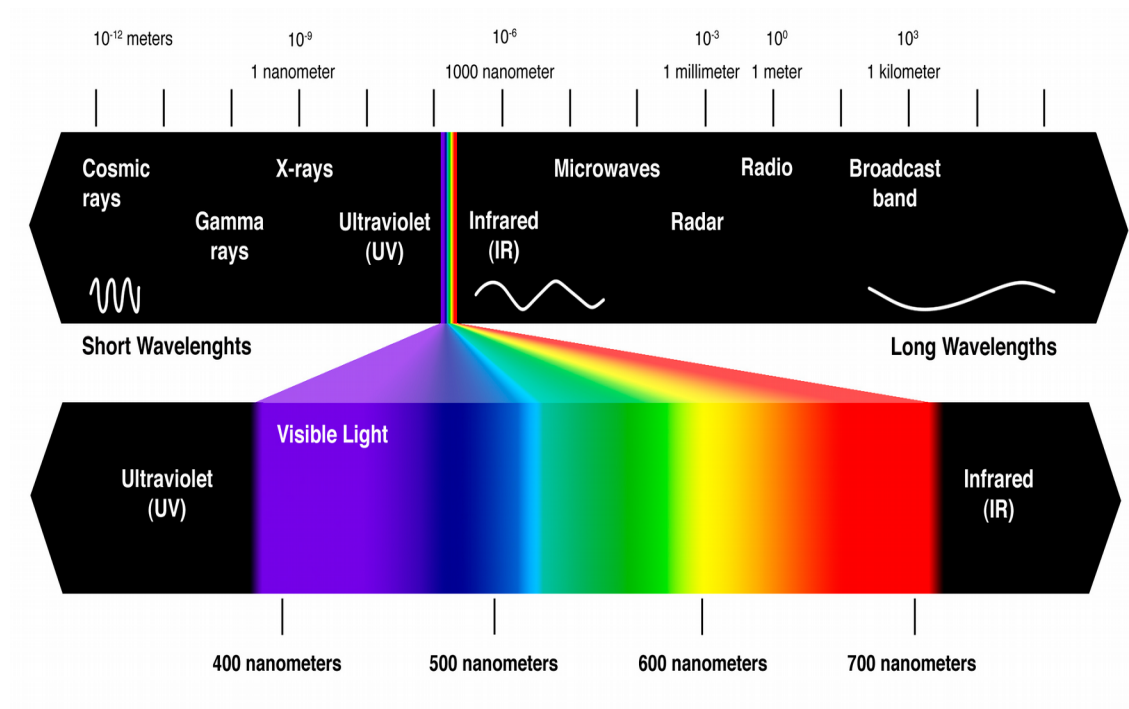


Figura 4.2: Espectro visible de la luz

4.1.3 El espacio de color RGB

Es uno de los espacios de color más utilizados. Es un sistema que se basa en la capacidad que tienen los conos del ojo humano en distinguir entre los colores rojo, verde y azul. Es un sistema de color en el cual la suma de todos sus colores genera el color blanco, se le llama sistema aditivo.

Un sistema RGB se basa en el sistema bidimensional de representación de imágenes en blanco y negro. Consiste en una matriz cuyos valores de luminancia de cada pixel, están compuestos por un total de 256 valores (en un sistema de 8 bits); donde cada uno de estos valores representa las distintas tonalidades de grises, desde el valor 0 (negro) hasta el valor 255 (blanco).

En un espacio RGB se parte desde la misma idea, solo que aquí tenemos una matriz de tres dimensiones que se compone de tres planos correspondientes cada uno de ellos a los tres colores primarios: rojo, verde y

azul. Cada uno de estos planos, es como una imagen en blanco y negro, con la diferencia de que los valores de cada uno de los planos, corresponden con las distintas tonalidades de dicho color. La combinación de estos tres planos, permite la representación de el resto de colores.

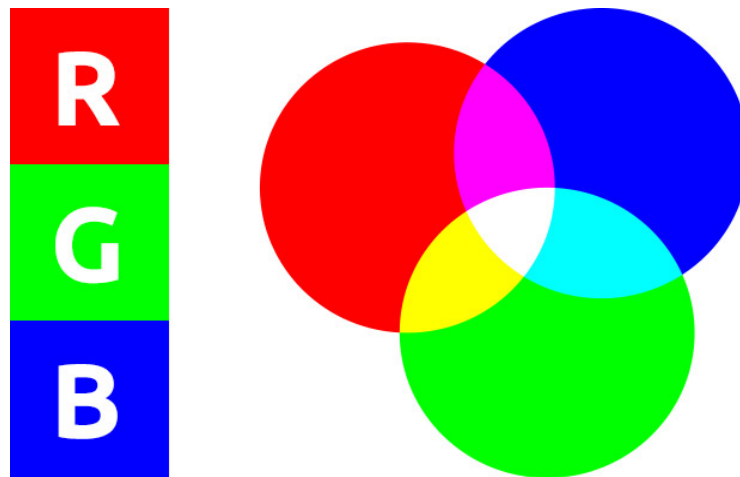


Figura 4.3: Colores RGB

Un valor de cero en todos los planos, representa la ausencia de color, es decir, el negro absoluto, mientras que un valor de 255 en todos los planos genera el blanco absoluto.

Cada uno de estos planos trabaja con 8 bits, por lo que si se tienen 256 valores diferentes, se obtienen un total de 16777216 posibles opciones de color.

4.1.4 Técnicas de pre-procesado de la imagen

El pre-procesado es una de las partes más importantes a la hora de tratar con imágenes digitales, ya que sirve para mejorarlas a la hora de obtener un resultado óptimo en cualquier tipo de problema. Estos procesos de mejora abarcan desde la realización de recortes para extraer regiones de interés, hasta técnicas de suavizado para eliminar ruidos en la imagen.

A continuación se van a mostrar distintas técnicas de pre-procesado:

Segmentación de la imagen:

En cualquier proceso en el que se requiera el reconocimiento de imágenes, uno de los pasos fundamentales es la diferenciación de los elementos que componen esta. Para ello, se aplican multitud de técnicas, dependiendo del tipo de imagen y los resultados que se quieran obtener de ella. Normalmente, la segmentación de imágenes monocroma, se basa en las características de los niveles de gris, tales como su discontinuidad y similitud. La discontinuidad busca zonas en las que ocurren cambios abruptos en los niveles de gris, mientras que la similitud busca regiones que se basen en las relaciones espaciales que pudiese haber entre los píxeles que forman dicha región.

A continuación se muestra una de las técnicas de segmentación más usada:

4.1.4.1 Histograma: Thresholding:

Es una representación de la cantidad de píxeles que hay en cada tonalidad de color de la imagen. Por ejemplo, en una imagen de tonos de gris se obtendría un diagrama de barras, donde la superficie de cada barra corresponde con la frecuencia de aparición de cada tonalidad de gris.



Figura 4.4: Histograma imagen en escala de grises

4.1.4.2 Umbralización:

La umbralización es uno de los más importantes métodos de segmentación. El objetivo es convertir una imagen en escala de grises a una nueva con sólo dos niveles, de manera que los objetos queden separados del fondo (ver *figura 4.5*).



Figura 4.5: Umbralización utilizando diferentes valores de umbral

4.1.5 Técnicas de post-procesado de la imagen

4.1.5.1 Procesado morfológico:

Normalmente, en la mayor parte de los casos de segmentación de imágenes, no se obtienen buenos resultados sobre los objetos o regiones segmentadas. En la mayoría de los casos aparecen píxeles con bordes imprecisos o discontinuos. Para solucionar este problema, se realiza un post-procesado que realce la geometría y la forma de los objetos de la imagen. Para

ello, se extrae de los objetos sus estructuras geométricas a través del elemento estructurante (*ver figura 4.6*). El tamaño y la forma de este tipo de elementos se seleccionan según los requisitos de la aplicación en particular y las formas que se quieran extraer.



Figura 4.6: Elementos estructurantes

4.1.5.2 Dilatación y erosión

La dilatación y erosión, son un tipo de operaciones morfológicas sobre la imagen. La erosión es un proceso que se encarga de comprobar si el elemento estructurante está contenido en un conjunto de píxeles. Si alguno de los píxeles por vecindad no cumpliese con la limitación marcada por el elemento estructurante, el valor del píxel central se pone a cero. Por lo que acaba disminuyendo el área del objeto sobre el que se ha aplicado, y la desaparición de los elementos que sean más pequeños que el estructurante. Una repetición continua de la erosión, conllevaría a la desaparición total de los objetos que contenga la imagen.

En resumen:

- Reduce bordes.
- Separa objetos próximos.
- Elimina puntos blancos separados.
- Amplía detalles negros pequeños



Figura 4.7: Ejemplo erosión

Por otra parte, la dilatación, es un proceso que se encarga de comprobar si el elemento estructurante esta contenido en al menos un único pixel. Si esto se cumple, el valor del píxel central se pone a uno. Por lo que se consigue la expansión del objeto en su región fronteriza y se cierran todas las discontinuidades que sean menores a el elemento estructurante. Una repetición continua de la operación de dilatación, conllevaría a una expansión completa de los objetos de la imagen, llegando a ocuparse entera.

En resumen:

- Amplía bordes.
- Une objetos próximos.
- Une puntos blancos próximos.
- Elimina los detalles negros pequeños.



Figura 4.8: Ejemplo de dilatación

4.2 Detección de movimiento en imágenes

En este punto, se va a considerar el problema de determinar la estimación del movimiento por medio de la formulación de la transformada de Fourier [8]. Se considera una sucesión $f(x, y, t), t=0, 1, \dots, T-1$, de T fotogramas de imágenes de tamaño $M \times N$ generadas por un dispositivo de captación de imágenes estacionario. Supongamos que todas las imágenes tienen un fondo estacionario de intensidad cero. La excepción se encuentra en un único objeto de 1 píxel de intensidad unidad, moviéndose con velocidad constante. Supóngase que para el fotograma uno ($t=0$) el plano imagen se proyecta sobre el eje x ; esto es, la intensidad de los píxeles se suman a lo largo de las columnas. Esta operación genera una matriz de M filas que son 0, a excepción de en la posición donde se ha proyectado el objeto. Al multiplicar los componentes de la matriz por $\exp[j2\pi k_1 x \Delta t], x=0, 1, \dots, M-1$, por el objeto de coordenadas (x', y') , en este instante de tiempo, se obtiene un resultado igual a $\exp[j2\pi k_1 \Delta t]$. En esta notación k_1 es un entero positivo, y Δt es el intervalo de tiempo entre dos fotogramas.

Supongamos que en el segundo fotograma ($t=1$), el objeto a realizado un movimiento horizontal a las coordenadas $(x'+1, y')$; es decir, se ha movido 1 píxel paralelo al eje x ; entonces, al repetir el proceso anterior se genera $\exp[j2\pi k_1(x'+t)\Delta t]$. Si el objeto continúa moviéndose 1 píxel por fotograma, entonces, en cualquier instante de tiempo, el resultado es $\exp[j2\pi k_1(x'+t)\Delta t]$, el cual utilizando la fórmula de Euler, se puede expresar como $\exp[j2\pi k_1(x'+t)\Delta t] = \cos[2\pi k_1(x'+t)\Delta t] + j \sin[2\pi k_1(x'+t)\Delta t]$ para $t=0, 1, \dots, T-1$. Es decir, este sistema genera una senoide compleja de frecuencia k_1 . Si el objeto se moviese v_1 píxeles, en la dirección x , entre fotogramas, la senoide podría tener la frecuencia $v_1 k_1$. Como varía entre 0 y (incremento enteros), la restricción de k_1 de que tenga valores enteros, da lugar a que la senoide compleja tenga picos en los extremos (y). El último pico se puede ignorar, ya que es debido a la simetría conjugada. Por lo que la búsqueda de un pico en el espectro de Fourier, da lugar a $v_1 k_1$ que al dividirlo por k_1 da lugar a v_1 que es la componente de la velocidad en la

dirección x . De igual forma, las proyecciones sobre el eje y darían v_2 que es la componente de la velocidad en la dirección y .

Si no se efectuasen movimientos en la sucesión de fotogramas, generaría términos idénticos, cuya transformada de Fourier produciría un único pico en la frecuencia 0. Por esto, dado que las operaciones realizadas son lineales, el caso general en el que existen uno o varios objetos en movimiento en un fondo estático, tendría una transformada de Fourier con un pico correspondiente a los componentes estáticos de la imagen y picos en las posiciones proporcionales a las velocidades de los objetos.

Resumiendo, para una sucesión de T imágenes de tamaño $M \times N$, la suma de las proyecciones ponderadas sobre el eje x en cualquier instante de tiempo es:

$$g_x(t, k_1) = \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y, t) e^{j2\pi k_1 x \Delta t}, t=0, 1, \dots, T-1 \quad (1)$$

De forma similar, la suma de las proyecciones sobre el eje y es:

$$g_y(t, k_2) = \sum_{y=0}^{N-1} \sum_{x=0}^{M-1} f(x, y, t) e^{j2\pi k_2 y \Delta t}, t=0, 1, \dots, T-1 \quad (2)$$

Las transformadas de Fourier unidimensionales de las ecuaciones (1) y (2) son:

$$G_x(u_1, k_1) = \frac{1}{T} \sum_{t=0}^{T-1} g_x(t, k_1) e^{-j2\pi u_1 t/T}, u_1=0, 1, \dots, T-1 \quad (3)$$

$$G_y(u_2, k_2) = \frac{1}{T} \sum_{t=0}^{T-1} g_y(t, k_2) e^{-j2\pi u_2 t/T}, u_2=0, 1, \dots, T-1 \quad (4)$$

En la práctica, el cálculo de estas transformadas se realiza utilizando un algoritmo de la FFT.

Capítulo 5. Restricciones

En este capítulo se van a exponer todas las restricciones, o limitaciones, existentes en el diseño y que van a condicionar la elección de una u otra alternativa. Estos factores restrictivos se van a estructurar en dos grupos, según su tipo, factores dato y factores estratégicos.

5.1 Factores dato

Las restricciones que se van a tratar aquí, son aquellas que no puedan ser modificadas durante el transcurso del proyecto, debido a su naturaleza o por petición del cliente.

- Una de las restricciones del proyecto es que la aplicación tiene que ser desarrollada en el lenguaje *C++* debido a su demostrada velocidad en el procesamiento de imágenes y a las ventajas derivadas de su naturaleza orientada a objetos que lo caracteriza.
- Debido a que se va a codificar en *C++* será necesario abordar el proceso de especificación de requisitos y el diseño de la aplicación desde la perspectiva de la orientación a objetos.
- Otra restricción viene dada en cuanto al sistema operativo de ejecución del software, que debe ser *GNU/Linux* esto se debe a la necesidad de usar librerías propias de dicho sistema operativo.
- En la medida de lo posible, los recursos software usados durante el desarrollo del proyecto, serán de carácter libre, debido a las

restricciones económicas que existen, al tratarse de un Proyecto Fin de Grado.

- Los recursos hardware también se deben limitar a aquellos propios del autor de proyecto, y las ayudas que estén disponibles por parte de los directores del proyecto.

5.2 Factores estratégicos

Las restricciones siguientes corresponden con las decisiones que se han tenido que tomar a la hora de desarrollar el proyecto.

- Para la grabación del audio, las diapositivas de la presentación y la conversión final del vídeo a un formato apropiado para ser compartido, se usarán las funcionalidades que aportan el programa *avconv*[9]. Dicho programa es un conversor de audio y vídeo, rápido y de alta calidad; el cual hace uso de la librería *libav*[3]. Por otra parte, se podría utilizar el programa *ffmpeg*[15] el cual es una colección de software libre que puede grabar, convertir y hacer streaming de audio y vídeo. Sin embargo, se ha preferido escoger *avconv* debido a que está basado en *ffmpeg* y tiene mayor soporte en las principales distribuciones de *GNU/Linux*.
- Para la codificación se va a trabajar con el editor de código *Atom*[2], para la parte del *backend*, ya que incorpora las herramientas para codificar de forma sencilla. Se podría escoger cualquier otro editor de texto plano, e incluso un IDE (*Entorno de Desarrollo Integrado*) como *Eclipse*[16] pero se ha preferido escoger *Atom* para la realización de la labor de la codificación debido a su fácil manejo y sencillez. Para la parte del *frontend* se va a utilizar el programa *Qt Creator*[5] el cuál se utilizará para realizar la interfaz gráfica en *Qt*. Existen otras bibliotecas para desarrollar interfaces gráficas como *GTK+* (*The GIMP Toolkit*) [17] pero se ha decidido usar *Qt* debido a que existen aplicaciones y

entornos de escritorio que han usado esta librería para desarrollar sus interfaces gráficas y han demostrado ser más rápidas y tener un gran diseño. Además *Qt* funciona en todas las plataformas principales, y posee un amplio apoyo, tanto de empresas como de usuarios. *Qt* no solo permite crear la interfaz gráfica sino que además ofrece toda una API de desarrollo, con métodos para acceder a bases de datos mediante SQL, uso de XML, gestión de hilos, soporte de red, etc.

- Para la detección del puntero y el procesamiento de las imágenes captadas por la *Webcam* se va a utilizar la librería *OpenCV*[4] ya que aporta los métodos necesarios para el cometido que acaece en este proyecto y prima en su estabilidad y seguridad. Existen otro tipo de librerías de visión por computador como *BoofCV*[18] pero se ha decidido usar *OpenCV* debido a que está desarrollado en *C++* al igual que la aplicación que se va a desarrollar y existen muchas aplicaciones de visión artificial desarrolladas con él y han demostrado su calidad, robustez y fiabilidad. Por otra parte, también se va a hacer uso de la rama de desarrollo *contrib*[19] de *OpenCV*, la cual incluye algunos de los métodos que se van a implementar en la aplicación.
- Para la elaboración de la documentación se utilizará el editor de textos *LibreOffice Writer*[10], así como el gestor de bibliografía *Zotero*[11] y el editor de diagramas *online draw.io*[12]. No se han tenido en cuenta otras alternativas a la edición de texto, como *Microsoft Office*, debido a que se desea realizar la mayor parte de este proyecto con software libre.

Capítulo 6. Recursos

6.1 Recursos humanos

El autor de este proyecto encargado del análisis, diseño, programación y de la elaboración de la documentación es el alumno del Grado en Ingeniería Informática Rafael Ulises Baena Herruzo.

Los directores encargados de la coordinación del proyecto son:

- D. Juan María Palomo Romero: Colaborador Honorario del Área de Conocimiento de Proyectos de Ingeniería y adscrito al Departamento de Ingeniería Rural de la Universidad de Córdoba.
- Dr. Lorenzo Salas Morera : Profesor Titular de Universidad del Área de Conocimiento de Proyectos de Ingeniería y adscrito al Departamento de Ingeniería Rural de la Universidad de Córdoba.

6.2 Recursos hardware

Para la Elaboración de la documentación se utilizarán dos ordenadores personales con las siguientes características:

Ordenador de sobremesa:

- Procesador *Intel® Core™ i7-6700* CPU - 3,40GHz
- 16,0 GB de memoria RAM

- 250 GB de disco duro en estado sólido (*SSD*).
- Tarjeta gráfica *NVIDIA GeForce GTX 960* – 4 GB *GDDR5*.

Ordenador portátil *HP Pavilion 15-n014ss*:

- Procesador *Intel® Core™ i7-4500U* CPU - 1,8GHz
- 8,0 GB de memoria RAM
- 120 GB de disco duro en estado sólido (*SSD*).
- Tarjeta gráfica *NVIDIA GeForce GT 740 M* – 2 GB *DDR3*.

Igualmente, se hará uso de una impresora a color *EPSON XP 202* para la impresión de la documentación, así como de dispositivo USB extraíble de 32 GB para el almacenamiento y transporte de la documentación.

En el Desarrollo del proyecto se utilizarán los mismos ordenadores personales indicados anteriormente.

Para la Implementación del proyecto, será necesaria una cámara web para capturar el vídeo. En este caso se utilizará una cámara web *logitech*.

6.3 Recursos software

El Sistema Operativo que se utilizará será *Linux Mint 18.1 Serena*[13], tanto para la elaboración como para el desarrollo del proyecto y su implementación.

En la Elaboración de la documentación, se utilizará como editor de texto *LibreOffice Writer versión 5.2.5.1*[10].

Para el Desarrollo del proyecto, se utilizará la biblioteca libre de visión artificial *OpenCV*[4] y programación en *C++*[1] utilizando el software *Atom*[2] para el *backend* y *Qt Creator*[5] para el *frontend*.

Capítulo 7. Especificación de requisitos

El proyecto que se pretende abordar, consiste en el desarrollo de una aplicación que permita el seguimiento de un puntero láser físico, durante una presentación de diapositivas, y representarlo en una grabación digital. También se desea, que la salida generada pueda ser utilizada para su visionado en plataformas dedicadas a compartir videos y con cualquier otro software.

En este apartado se detallan todos los requisitos que especifiquen lo que se espera que realice el sistema, lo que debe hacer para cumplirlos, la información con la que va a tratar, lo que deben satisfacer los componentes externos que interactúan con el usuario y las característica que éste deberá presentar, pero que no añadan ninguna funcionalidad.

7.1 Requisitos de información

En este apartado se recoge la información que la aplicación deberá gestionar y almacenar. Dichos requisitos van a ser codificados con las siglas RI seguidas de un número de identificación.

Los requisitos de información son:

- **RI-01:** Información sobre la posición del puntero láser.

- ◆ Descripción: el sistema almacenará la información relativa a la posición del puntero láser en un instante de tiempo.
- ◆ Datos específicos: los datos que se manejarán son:
 - Coordenada (x, y) de la posición del puntero láser.
 - Tiempo en microsegundos.
- **RI-02:** Captura del audio de la presentación.
 - ◆ Descripción: el sistema almacenará el audio generado durante la presentación.
 - ◆ Datos específicos:
 - Formato de compresión de audio MP3 (*MPEG Audio Layer III*).
- **RI-03:** Captura de las diapositivas de la presentación.
 - ◆ Descripción: el sistema almacenará en un fichero de vídeo las diapositivas de la presentación en formato digital.
 - ◆ Datos específicos:
 - Formato de compresión de vídeo *H.264*[14].
- **RI-04:** Archivo de vídeo final.
 - ◆ Descripción: el sistema almacenará en un fichero de vídeo el montaje final con el audio grabado y las capturas de las diapositivas de la presentación, con el puntero láser representado en estas.
 - ◆ Datos específicos:
 - Formato de compresión de vídeo *H.264*[14].
 - Formato de compresión de audio MP3 (*MPEG Audio Layer III*).

7.2 Requisitos funcionales

A continuación se van a detallar los requisitos funcionales que deberá satisfacer el sistema software. Dichos requisitos van a ser codificados con las siglas RF seguidas de un número de identificación.

Los requisitos funcionales son:

- **RF-01:** El sistema deberá permitir al usuario la selección de una cámara, del conjunto total de dispositivos de captación de imágenes conectados al ordenador.
- **RF-02:** El sistema deberá permitir al usuario cargar un archivo, el cuál contenga la presentación ya grabada, en el caso de que el usuario no quiera realizar la captura en directo.
- **RF-03:** El sistema deberá permitir al usuario previsualizar el contenido de cada una de las cámaras conectadas.
- **RF-04:** El sistema deberá permitir al usuario seleccionar un dispositivo de captación de sonido, entre todos los micrófonos conectados al ordenador.
- **RF-05:** El sistema deberá permitir al usuario poder solicitar el inicio de la captura de la presentación (obtención de las coordenadas del puntero láser, grabación de las diapositivas y del audio).
- **RF-06:** El sistema deberá permitir al usuario la selección de la superficie de proyección seleccionando esta con el cursor del ordenador.
- **RF-07:** El sistema deberá mostrar al usuario, las imágenes captadas por la cámara y un objeto circular rodeando al puntero láser, así se podrá cerciorar de la correcta captación de este.
- **RF-08:** El sistema deberá permitir al usuario finalizar la captura de la presentación.
- **RF-09:** El sistema deberá permitir al usuario seleccionar la ruta donde se guardará el archivo de vídeo con el montaje final.
- **RF-10:** El sistema deberá ser capaz de mostrar a el usuario el progreso del montaje.
- **RF-11:** El sistema deberá permitir al usuario poder iniciar el montaje de la presentación.

- **RF-12:** El sistema deberá permitir al usuario poder seleccionar la relación de aspecto de la presentación.
- **RF-13:** El sistema deberá permitir al usuario utilizar todos los modos de pantalla disponibles por el ordenador, siendo estos, pantalla espejada, mostrar solo en segunda pantalla y pantalla extendida.
- **RF-14:** El sistema deberá permitir al usuario la utilización del modo pantalla extendida del ordenador, para ello facilitará la introducción de la resolución de la pantalla principal por parte del usuario.
- **RF-14:** El sistema deberá permitir al usuario la realización de un montaje *offline*.
- **RF-15:** Si se ha seleccionado un montaje *offline*, el sistema deberá permitir al usuario la selección de la ruta donde se guardarán y cargarán los archivos de audio y vídeo generados durante la presentación y el archivo de captura de las coordenadas del puntero láser.

7.3 Requisitos no funcionales

A continuación se van a detallar los requisitos no funcionales que deberá satisfacer el sistema. Dichos requisitos van a ser codificados con las siglas RNF seguidas de un número que lo identifique.

Los requisitos no funcionales son:

- **RNF-01:** La aplicación deberá ser desarrollada como aplicación de escritorio.
- **RNF-02:** El sistema deberá ser eficaz, robusto y fiable.
- **RNF-03:** El sistema se deberá poder mantener de forma fácil.
- **RNF-04:** La interfaz de usuario de la aplicación, deberá ser lo más intuitiva posible, siendo de fácil entendimiento y manejo, para que cualquier usuario de esta pueda hacer un uso correcto de la aplicación.

- **RNF-05:** El sistema deberá responder en un tiempo lo suficientemente aceptable a las peticiones de los usuarios.
- **RNF-06:** El sistema deberá responder contra los posibles errores, ya sean por parte del usuario o del propio sistema.
- **RNF-07:** Todos los mensajes de error deberán ser significativos, de tal forma que incluyan un texto descriptivo y indicaciones que el usuario deberá efectuar ante dicho error.
- **RNF-08:** El sistema deberá estar siempre disponible.

Capítulo 8. Análisis del sistema

8.1 Análisis de los casos de uso

En este apartado se van a describir, mediante los diagramas de casos de uso, el comportamiento y la comunicación que deberá tener el sistema mediante la interacción con el usuario pero sin llegar a especificar cómo se va a implementar dicho comportamiento.

Para especificar los casos de uso se seguirá la plantilla que se puede observar en la *Tabla 8.1*.

Casos de uso	Nombre del caso de uso.
Actores	Actores que intervienen en el caso de uso.
Descripción	Descripción informar de los objetivos del caso de uso.
Precondiciones	Conjunto de precondiciones que deben cumplirse para que el caso de uso se realice de forma correcta.
Pasos a seguir	Secuencia de pasos necesarios a seguir para la correcta realización del caso de uso.

Tabla 8.1: Plantilla de casos de uso

A la vista de los objetivos y requisitos que se han ido exponiendo en el sistema a desarrollar, sólo habrá un tipo de actor, que representará al usuario de la aplicación.

En los siguientes apartados se van a especificar los casos de uso de la aplicación que se va a desarrollar.

8.1.1 Caso de uso 0: Sistema de captura y representación de puntero láser

En la *Figura 8.1* se pretende mostrar una visión global de la funcionalidad que el sistema debe proporcionar a los actores del mismo, que posteriormente se irán desgranando para dar mayor nivel de detalle de cada una de las funcionalidades.

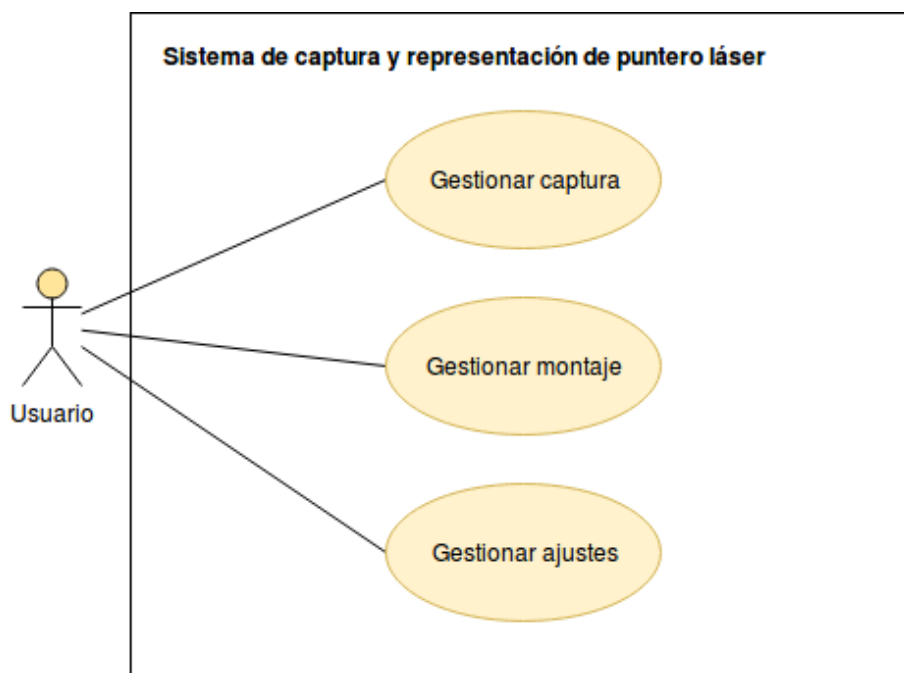


Figura 8.1: Diagrama de caso de uso 0: Sistema de captura y representación de puntero láser

8.1.2 Caso de uso 1: Gestionar captura

Este caso de uso permite llevar a cabo las diferentes operaciones relacionadas con la gestión de la captura del puntero láser y el audio y vídeo de la presentación.

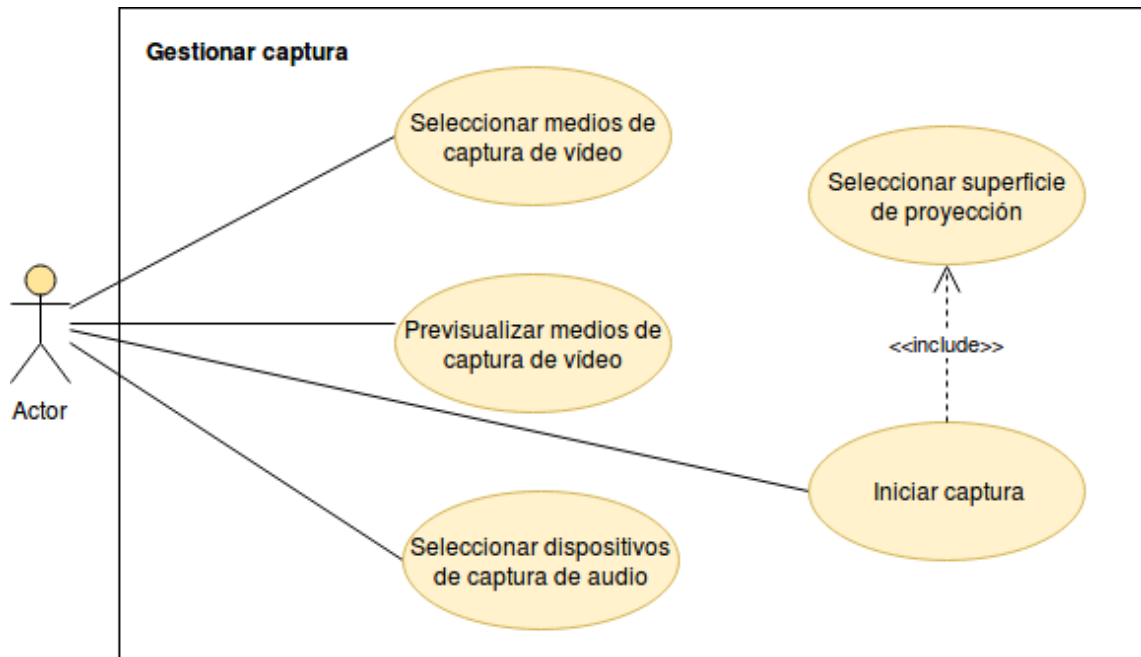


Figura 8.2: Diagrama de caso de uso 1: Gestionar captura

El conjunto de casos de uso asociados a la gestión de captura es el siguiente:

- Seleccionar medios de captura de vídeo. El sistema mostrará dos opciones para seleccionar la fuente de captura de vídeo, una para cargar un archivo de vídeo y otra para poder seleccionar un dispositivo de captación de imágenes conectado al ordenador.

Casos de uso	Seleccionar medios de captura de vídeo.
Actores	Usuario.
Descripción	El sistema muestra al usuario dos opciones. En la primera podrá seleccionar un archivo de vídeo como fuente de captura. En la segunda podrá seleccionar un dispositivo de captura de vídeo conectado al ordenador.
Precondiciones	El archivo de vídeo debe ser correcto si se ha elegido dicha opción, o el dispositivo de captura de vídeo seleccionado debe existir.
Pasos a seguir	Se tienen dos posibles pasos a seguir: <ol style="list-style-type: none"> 1. El usuario selecciona buscar un archivo de captura. <ol style="list-style-type: none"> 1.1. El sistema le muestra una ventana de selección de archivos. 1.2. El usuario selecciona el archivo. 2. El usuario selecciona el identificador de el dispositivo de captación de vídeo.

Tabla 8.2: Caso de uso 1.1: seleccionar medios de captura de vídeo

- Previsualizar medios de captura de vídeo. El sistema dará al usuario la opción de poder previsualizar el contenido de la cámara o el archivo seleccionado.

Casos de uso	Previsualizar medios de captura de vídeo.
Actores	Usuario.
Descripción	El sistema mostrará al usuario una imagen correspondiente al dispositivo o archivo de captura seleccionado previamente.
Precondiciones	Para que el sistema muestre la previsualización, el usuario debe haber seleccionado un medio de captura de vídeo.
Pasos a seguir	<ol style="list-style-type: none">1. El usuario selecciona la opción de previsualización.2. El sistema muestra una imagen del dispositivo o archivo de captura seleccionado.

Tabla 8.3: Caso de uso 1.2: previsualizar medios de captura de vídeo

- Seleccionar dispositivo de captura de audio. El sistema mostrará los distintos dispositivos hardware de captura de audio disponibles para su selección por parte del usuario.

Casos de uso	Seleccionar dispositivo de captura de audio.
Actores	Usuario.
Descripción	El sistema mostrará los distintos dispositivos hardware de captura de audio disponibles para su selección por parte del usuario.
Precondiciones	El dispositivo de captura de audio debe existir.
Pasos a seguir	<ol style="list-style-type: none">1. El sistema muestra una lista con el conjunto de dispositivos de captura de audio disponibles.2. El usuario selecciona el identificador del dispositivo de captura de audio.

Tabla 8.4: Caso de uso 1.3: seleccionar dispositivo de captura de audio

- Iniciar captura. El sistema iniciará la captura del audio y diapositivas de la presentación y del puntero láser.

Casos de uso	Iniciar captura.
Actores	Usuario.
Descripción	El sistema iniciará la captura del audio y diapositivas de la presentación y del puntero láser.
Precondiciones	Se debe haber seleccionado previamente el dispositivo de captura de audio y de vídeo.
Pasos a seguir	<ol style="list-style-type: none"> 1. El usuario selecciona la opción iniciar captura. 2. Incluye << Seleccionar superficie de proyección >>

Tabla 8.5: Caso de uso 1.4: iniciar captura

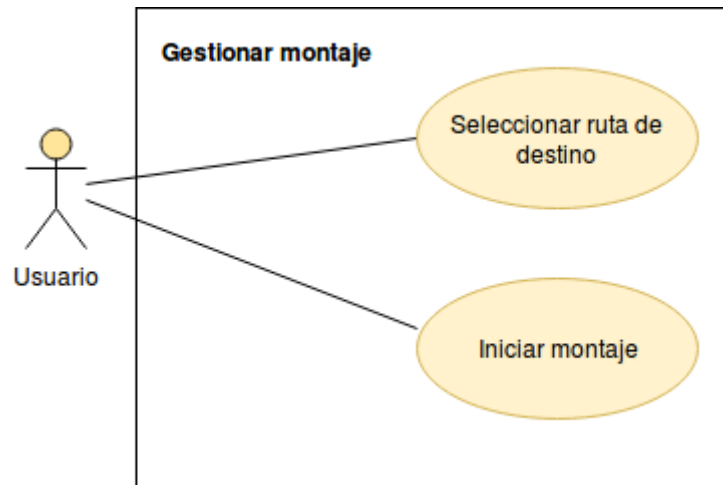
- Seleccionar superficie de proyección. El sistema le pedirá al usuario que seleccione la superficie donde se van a proyectar las diapositivas.

Casos de uso	Seleccionar superficie de proyección.
Actores	Usuario.
Descripción	El usuario selecciona la superficie donde se proyectarán las diapositivas.
Precondiciones	Haber iniciado la captura.
Pasos a seguir	<ol style="list-style-type: none"> 1. El usuario selecciona la superficie de proyección. 2. El sistema inicia la captura.

Tabla 8.6: Caso de uso 1.5: seleccionar superficie de proyección

8.1.1 Caso de uso 2: Gestionar montaje

Este caso de uso permite llevar a cabo las diferentes operaciones relacionadas con la gestión del montaje de la captura previamente realizada en el caso de uso 1.



*Figura 8.3: Diagrama de caso de uso
2: Gestionar montaje*

El conjunto de casos de uso asociados a la gestión del montaje es el siguiente:

- Seleccionar ruta de destino. El usuario deberá seleccionar la ruta donde se guardará el archivo de vídeo final.

Casos de uso	Seleccionar ruta de destino.
Actores	Usuario.
Descripción	El usuario selecciona la ruta y el nombre del archivo de salida del montaje.
Precondiciones	Que no exista el archivo seleccionado y exista el directorio seleccionado.
Pasos a seguir	<ol style="list-style-type: none">1. El usuario selecciona la ruta del directorio donde se guardará el montaje.2. El usuario introduce el nombre del archivo de salida.

Tabla 8.7: Caso de uso 2.1: seleccionar ruta de destino

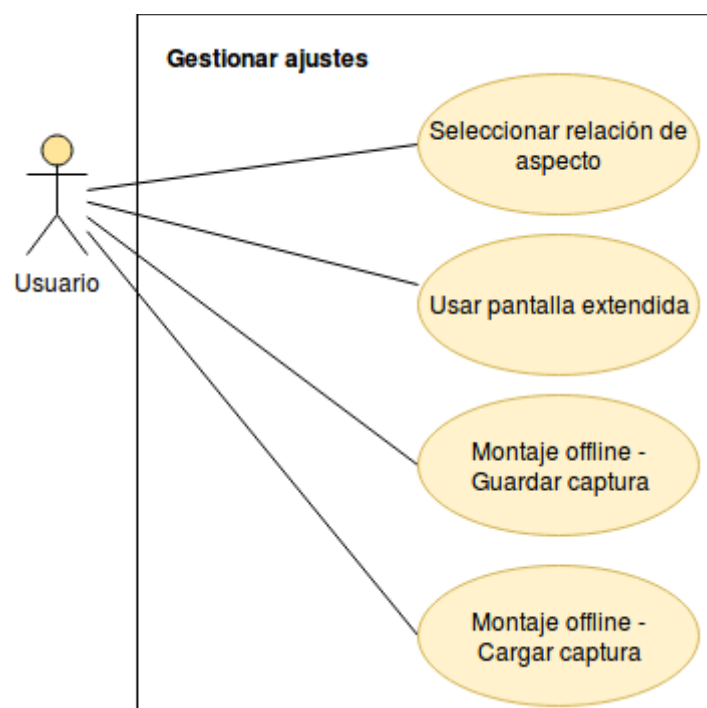
- Iniciar montaje. El sistema realizará el montaje final, uniendo el audio y las diapositivas capturadas con la representación virtual del puntero láser.

Casos de uso	Iniciar montaje.
Actores	Usuario.
Descripción	El usuario selecciona la opción de iniciar montaje. El sistema realiza el montaje de la captura previamente realizada.
Precondiciones	Que se haya configurado una ruta de salida y que se haya realizado la captura de la presentación, especificada en el caso de uso 1.
Pasos a seguir	<ol style="list-style-type: none">1. El usuario selecciona la opción de iniciar montaje.2. El sistema inicia el montaje y muestra el progreso de este.

Tabla 8.8: Caso de uso 2.2: iniciar montaje

8.1.2 Caso de uso 3: Gestionar ajustes

Este caso de uso permite llevar a cabo las diferentes operaciones relacionadas con la gestión de los distintos ajustes del programa. Estos ajustes están relacionados tanto con la captura de la presentación como con el montaje de esta.



*Figura 8.4: Diagrama de caso de uso
3: Gestionar ajustes*

El conjunto de casos de uso asociados a la gestión de los ajustes es el siguiente:

- Seleccionar relación de aspecto. El usuario podrá seleccionar la relación de aspecto que tiene la presentación o bien dejarlo en automática.

Casos de uso	Seleccionar relación de aspecto.
Actores	Usuario.
Descripción	El usuario selecciona la relación de aspecto que tienen las diapositivas que se están proyectando o selecciona su detección automática.
Precondiciones	-
Pasos a seguir	<ol style="list-style-type: none"> 1. El usuario selecciona la relación de aspecto que tienen las diapositivas. 2. El sistema guarda la configuración.

Tabla 8.9: Caso de uso 3.1: seleccionar relación de aspecto

- Usar pantalla extendida. El usuario podrá seleccionar la opción de usar la pantalla de su ordenador de forma extendida.

Casos de uso	Usar pantalla extendida.
Actores	Usuario.
Descripción	El usuario selecciona la opción de usar pantalla extendida.
Precondiciones	El ordenador debe estar conectado al proyector con la opción de pantalla extendida.
Pasos a seguir	<ol style="list-style-type: none"> 1. El usuario introduce el ancho y alto de su pantalla principal. 2. El sistema guarda la configuración.

Tabla 8.10: Caso de uso 3.2: usar pantalla extendida

- Montaje offline – guardar captura. El usuario podrá seleccionar la realización de un montaje offline, para ello deberá indicar donde se guardarán los archivos capturados.

Casos de uso	Montaje offline – guardar captura.
Actores	Usuario.
Descripción	El usuario selecciona la opción de realizar un montaje offline, guardar captura. Para ello se guardarán los archivos capturados en la ruta indicada por el usuario.
Precondiciones	Se debe realizar antes de la captura y el montaje.
Pasos a seguir	<ol style="list-style-type: none">1. El usuario introduce el directorio donde guardar los archivos.2. El usuario introduce el nombre de los archivos de audio, vídeo y captura.3. El sistema guarda la configuración.

Tabla 8.11: Caso de uso 3.3: montaje offline – guardar captura

- Montaje offline – cargar captura. El usuario podrá seleccionar la realización de un montaje offline, para ello deberá indicar donde se encuentran los archivos previamente capturados.

Casos de uso	Montaje offline – cargar captura.
Actores	Usuario.
Descripción	El usuario selecciona la opción de realizar un montaje offline, cargar captura. Para ello se cargarán los archivos guardados previamente por el usuario.
Precondiciones	Se debe realizar antes del montaje. La captura debe estar realizada y los archivos deben ser correctos.
Pasos a seguir	1. El usuario selecciona los archivos. 2. El sistema guarda la configuración.

Tabla 8.12: Caso de uso 3.4: montaje offline – cargar captura

8.2 Diagramas de secuencia

Los diagramas de secuencia se utilizan para describir el comportamiento de un sistema. Estos diagramas describen la colaboración que existe entre los objetos que componen el sistema. El diagrama de secuencias consta de objetos que se representan del modo usual: rectángulos con nombre subrayado, mensajes representados por líneas continuas con una punta de flecha y el tiempo representado como una progresión vertical[20].

8.2.1 Diagrama de secuencia 1: Gestionar captura

En la *Figura 8.5* se puede observar el diagrama de secuencia correspondiente con el *caso de uso 1: Gestionar captura*.

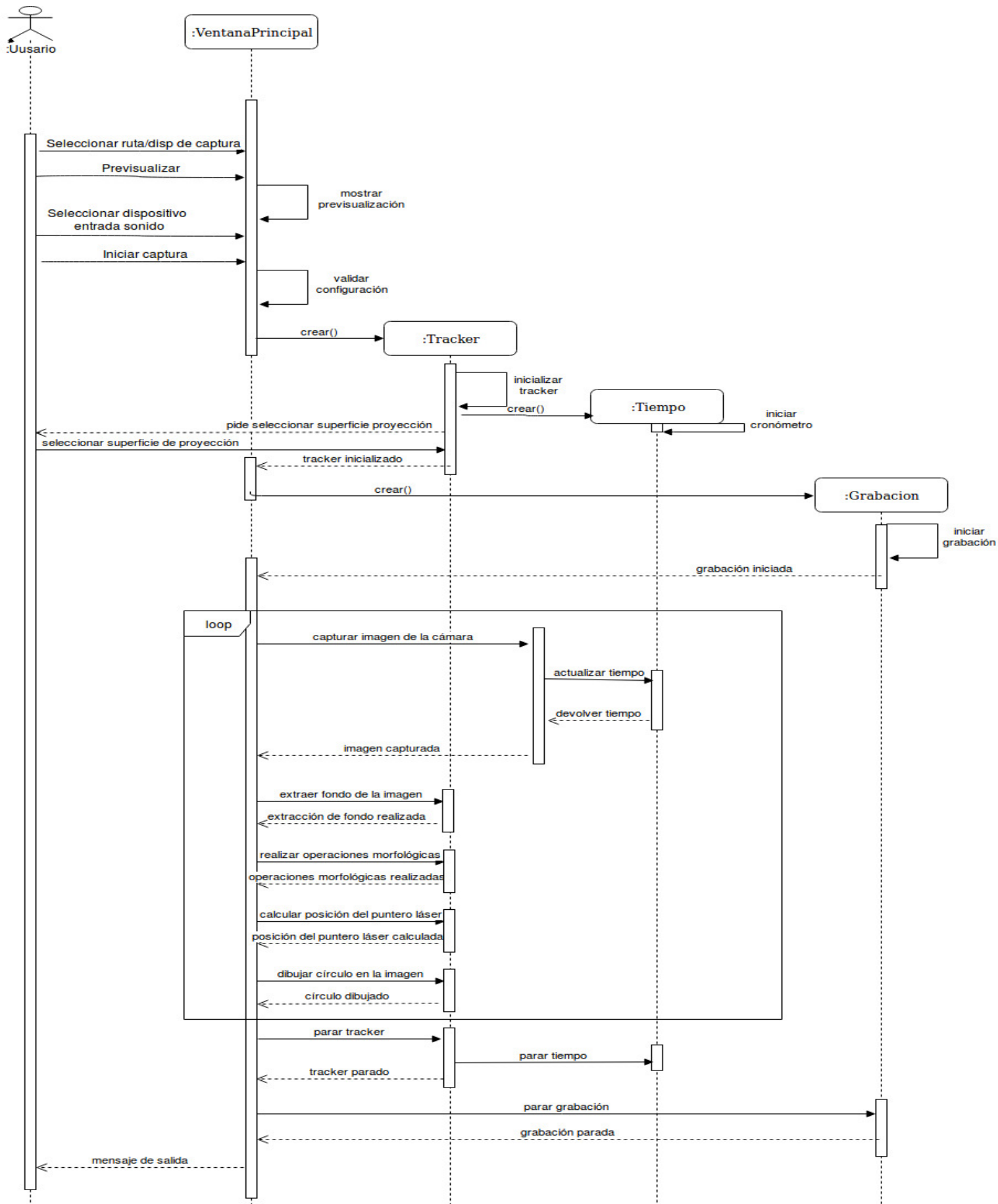


Figura 8.5: Diagrama de secuencia 1: Gestionar captura

En este caso el rol desempeñado por el único actor *Usuario*, es el que desencadena las acciones previas a la captura y el inicio de la captura del puntero láser y la grabación de audio y vídeo (correspondiente a la diapositivas que se están proyectando). Una vez finalizado, se tienen los archivos correspondientes a las grabaciones y las correspondientes coordenadas del puntero láser junto al tiempo en el que fueron captadas.

8.2.2 Diagrama de secuencia 2: Gestionar montaje

En la *Figura 8.6* se puede observar el diagrama de secuencia correspondiente con el *caso de uso 2: Gestionar montaje*.

En este caso el rol desempeñado por el Usuario, es el que desencadena las acciones de *seleccionar ruta de destino* e *iniciar montaje*.

Una vez finalizado el montaje, se tiene el archivo vídeo final, correspondiente a la grabación de las diapositivas con las coordenadas del puntero láser representadas en estas y el audio grabado durante la presentación. El archivo de vídeo se localizará en la ruta de destino previamente indicada por el usuario.

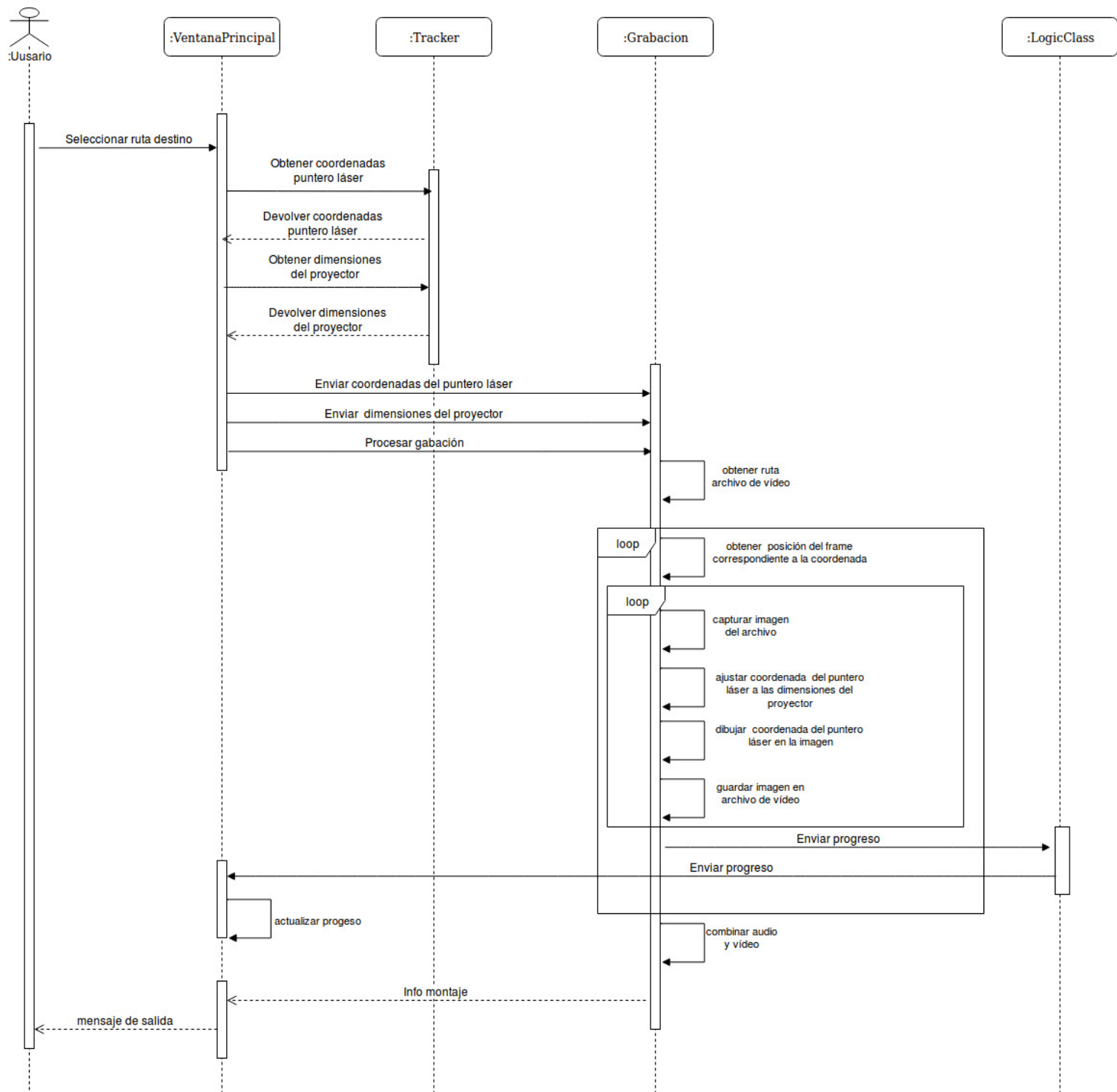


Figura 8.6: Diagrama de secuencia 2: Gestionar montaje

Capítulo 9. Especificación de la interfaz

9.1 Introducción

En este capítulo se van a definir las características que deberá tener la interfaz que utilizarán los usuarios del programa. Los requisitos que se especifican en este análisis van a dar como resultado el *Diseño de la Interfaz* que se detallará en el capítulo 10.

La interfaz es una de las partes más importantes de un Sistema Informático, ya que es lo que los usuarios van a ver del mismo y con el que van a interactuar, llevando a cabo una relación entre el usuario y el software.

La interfaz debe ser lo más simple y sencilla posible para que al usuario le resulten cómodas las distintas operaciones que desee realizar en el sistema, sin llegar a resultar complicado y tedioso. También cabe destacar, que la interfaz debe ser visualmente atractiva y que no aburra con su uso.

También se tendrán en cuenta otros aspectos como la legibilidad, el tamaño y color de las fuentes de texto, además de su contraste con el fondo, los cuales son factores que facilitarán la lectura de los textos.

Finalmente, se le otorgarán nombres descriptivos a los elementos que aparecerán, para facilitar el entendimiento al usuario. Además se incluirá una breve descripción de las acciones que realiza cada campo, de esta forma si el usuario no entiende en algún momento que acciones se van a realizar o no sabe que hacer, simplemente tendrá que colocar el ratón encima de estos.

9.2 Característica de la interfaz

La interfaz constará de una única ventana principal en la cual se encontrarán distintas pestañas para acceder a las distintas funcionalidades del sistema.

En el diseño de la ventana principal, aparecerán cuatro pestañas bien definidas tal como se puede observar en la *Figura 9.1*.

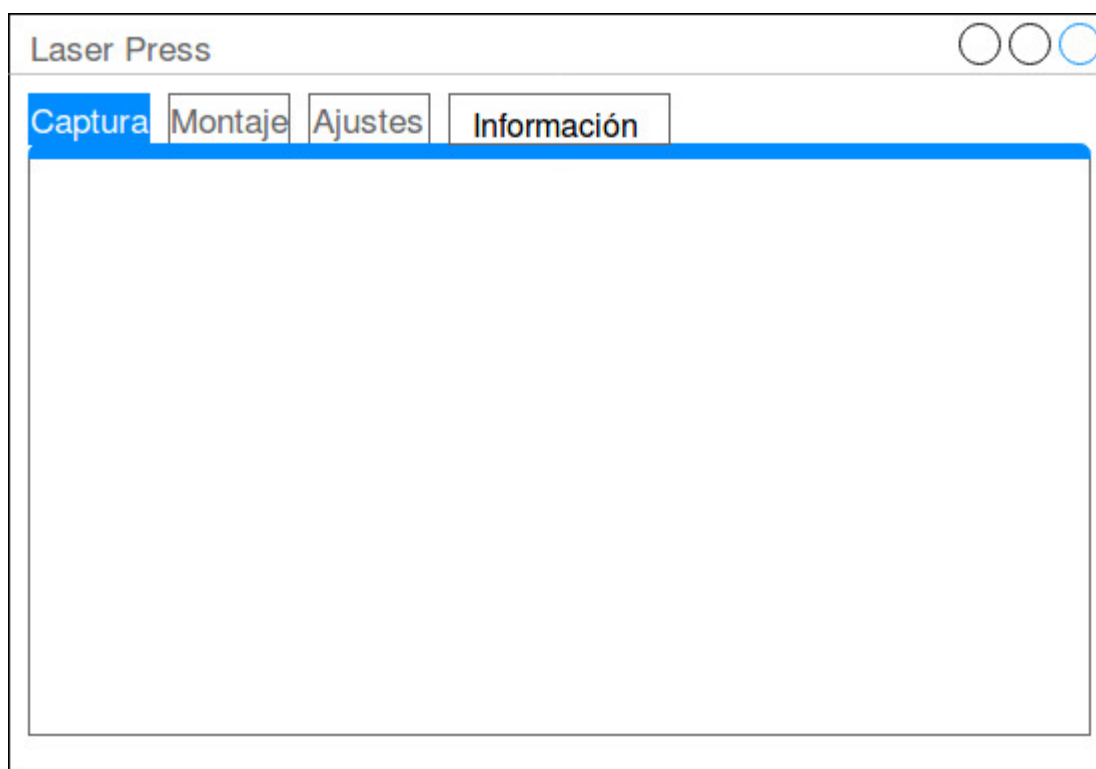


Figura 9.1: Estructura de pestañas de la ventana principal de la interfaz

9.2.1 Pestaña captura

En esta pestaña se podrán realizar todas las acciones referentes a la captura del puntero láser y la grabación de la presentación. En la *Figura 9.2* se puede observar la estructura que tendrá esta pestaña.

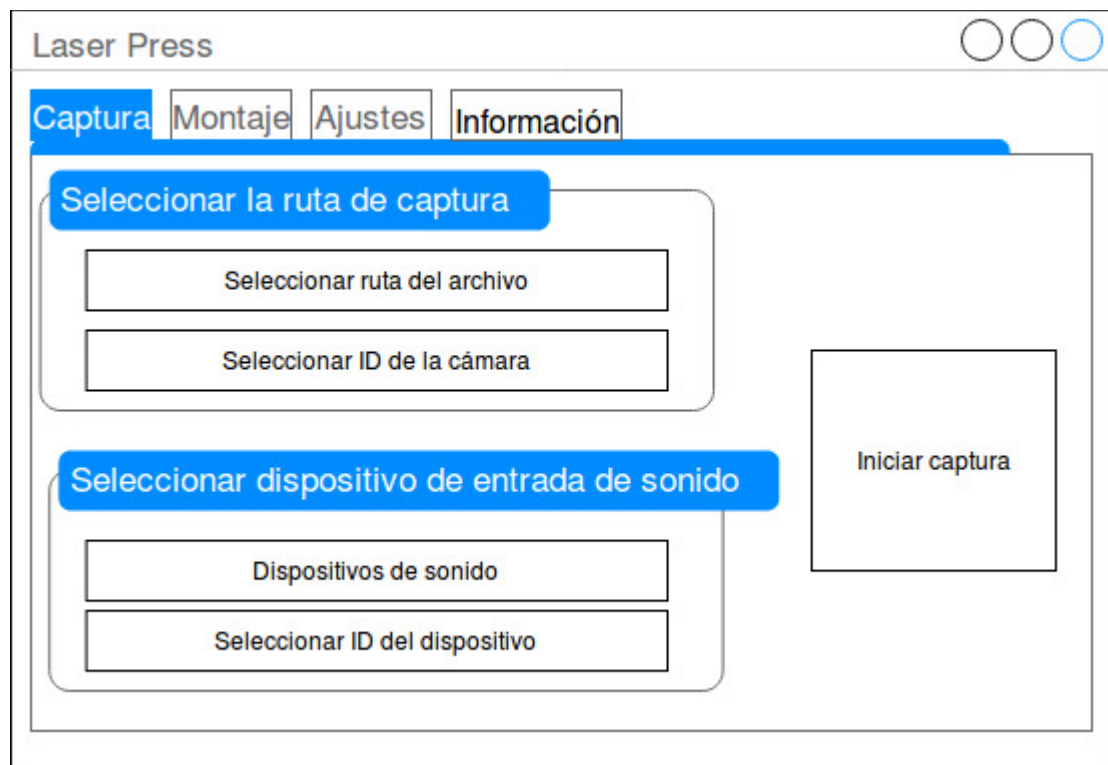


Figura 9.2: Estructura de la pestaña captura de la interfaz

Como se puede ver, esta pestaña se caracteriza por tres secciones principales:

- Selección de la ruta de captura del puntero láser, en la cual se podrá seleccionar un archivo o una cámara conectada al ordenador
- Selección de dispositivo de entrada de sonido, en la cual se mostrarán los distintos dispositivos de audio conectados al ordenador y el *usuario* podrá seleccionar el dispositivo correspondiente.
- Iniciar captura, la cuál será la parte encargada de iniciar la captura de la presentación.

9.2.2 Pestaña montaje

En esta pestaña se podrán realizar todas las acciones referentes al montaje de la captura realizada previamente. En la *Figura 9.3* se puede observar la estructura que tendrá esta pestaña.

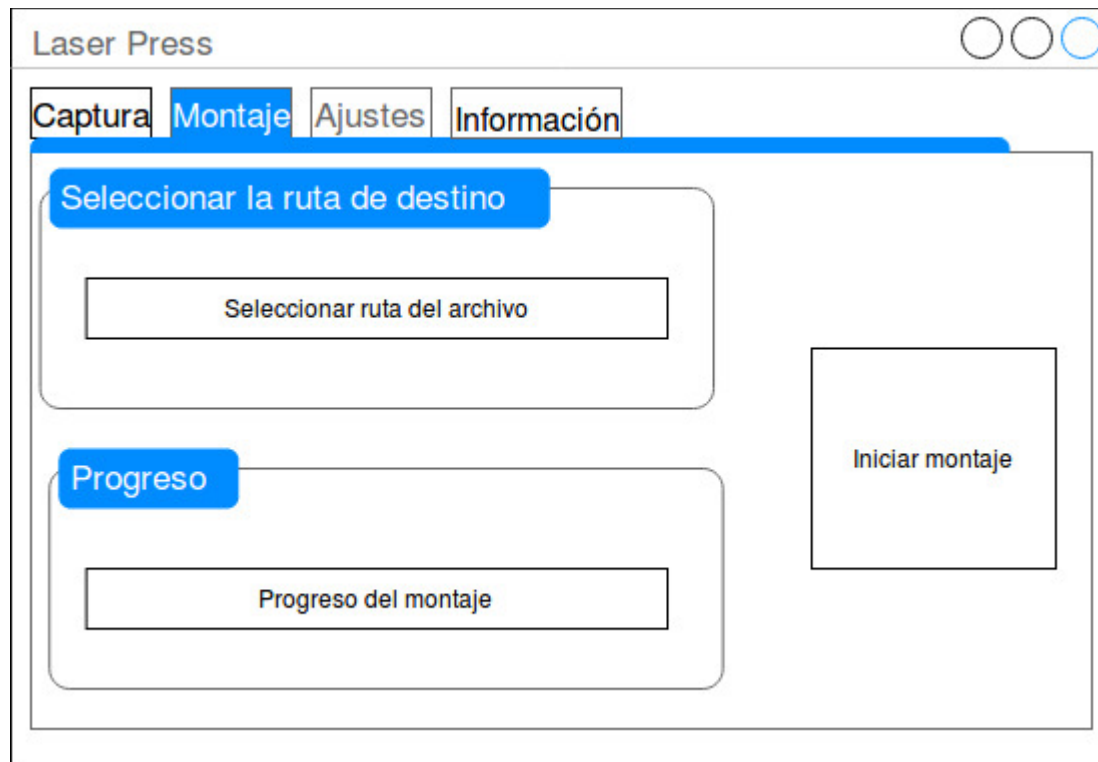


Figura 9.3: Estructura de la pestaña montaje de la interfaz

Como se puede ver, esta pestaña se caracterizará por tres secciones principales:

- Selección de ruta de destino, en la cual se podrá seleccionar la ruta donde guardará el archivo final de vídeo.
- Progreso, en la cual se mostrará el progreso que lleve el proceso de montaje de la presentación.
- Iniciar montaje, la cuál será la parte encargada de iniciar el montaje, encargado de digitalizar las coordenadas del puntero láser y unir audio y vídeo.

9.2.3 Pestaña ajustes

En esta pestaña se podrán realizar todas las acciones referentes a los distintos ajustes de la captura y el montaje. En la *Figura 9.4* se puede observar la estructura que tendrá esta pestaña.

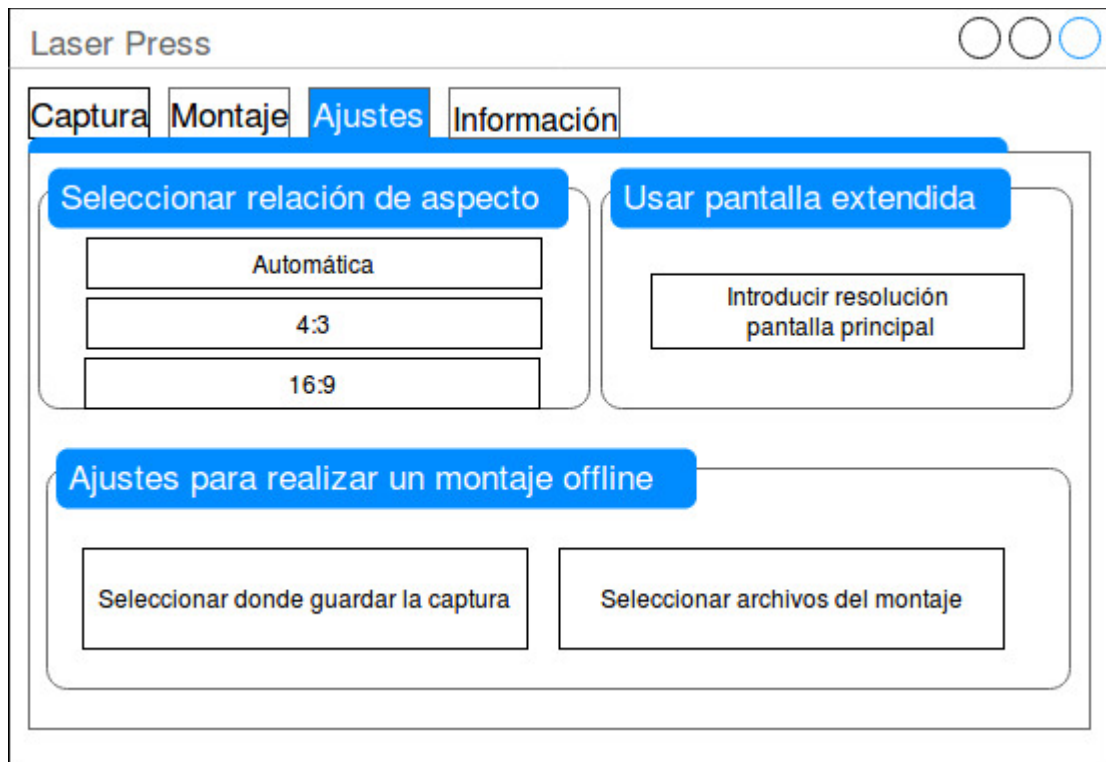


Figura 9.4: Estructura de la pestaña ajustes de la interfaz

Como se puede ver, esta pestaña se caracterizó por tres secciones principales:

- Selección de relación de aspecto, en la cual se podrá seleccionar la relación de aspecto que tienen las diapositivas al proyectarse sobre la superficie de proyección.
- Usar pantalla extendida, en la cual se mostrará elegir el uso de la pantalla del ordenador de forma extendida, para ello el *usuario* introducirá la resolución de su pantalla principal.

- Ajustes para realizar un montaje offline, la cuál se encargará de brindar al *usuario* las opciones para poder realizar un montaje *offline*, para ello, se deberá indicar donde se guardará la captura de la presentación antes de realizarla y posteriormente, una vez realizada la captura, se deberán indicar los archivos previamente guardados, para la realización del montaje.

En la pestaña restante, llamada *Información* se incluirá información sobre el programa y el autor de este.

Por último, existirán varios tipos de mensajes que se le presentarán al *usuario*, el sistema informático tendrá dos ventanas, una para los mensajes correctos (cuando una operación se haya realizado con éxito) o de tipo informativo y otra para los mensajes que informen de problemas en la ejecución del programa (operaciones que no se han podido realizar, alguna opción incorrecta, etc).

Capítulo 10. Diseño del sistema

En este capítulo se va a definir la arquitectura general del proyecto, especificando las partes que componen este y la relación entre ellos.

10.1 Diseño de datos

En este apartado se van a definir los distintos ficheros que usará el sistema software. Surge la necesidad de definir la estructura de dichos ficheros, por lo que a continuación se detallarán estos junto a su composición. Estas se van a extraer de la especificación de requisitos de información que se hizo del problema.

En cada uno de los ficheros, que se van a describir a continuación, se distinguirán principalmente los siguientes apartados:

- Nombre del archivo: nombre que se le va a asignar al archivo para su identificación.
- Descripción general: se explicará el objetivo u objetivos del fichero, de una forma breve y concisa.
- Estructura: se analizará la estructura del fichero, así como los campos de los que está formado si los tuviese.

10.1.1 Fichero de captura

- Nombre: *archivo.cap*

- Descripción: el sistema va a almacenar la información relativa a la posición del puntero láser en un instante de tiempo. Además almacenará la resolución de la pantalla de la que ha capturado dichas coordenadas. El nombre del archivo podrá ser el que el usuario le especifique, pero siempre manteniendo la extensión “.cap”.
- Estructura: la estructura del archivo será la siguiente:
 - La primera línea del fichero guardará la resolución de la superficie de proyección (la seleccionada por el usuario). Dicha resolución será guardada como un punto con sus correspondientes coordenadas (x, y) separadas por un espacio y representando estas el ancho y alto de la pantalla respectivamente.

Ejemplo: 640 480

- El resto de líneas del fichero corresponderán con la posición del puntero láser y el instante de tiempo (en microsegundos) en el que fueron captadas dichas coordenadas. El formato concreto que tendrá cada línea será, la coordenada (x,y) separada por un espacio y un número entero positivo.

Ejemplo:

80.5 38.5 79780

0 0 121175

309.738 231.738 161251

164 153 201819

39 252 242927

10.1.2 Fichero de audio

- Nombre: *audio.mp3*
- Descripción: el sistema almacenará el audio generado por la presentación, generalmente por el orador. El nombre de este archivo va a ser generado por el sistema informático de forma aleatoria, en caso de que no se especifique un montaje *offline*. Por otra parte, si el usuario

especificase un montaje *offline*, este, le deberá indicar un nombre a este archivo, para posteriormente poder hacer referencia a el.

- Estructura: la estructura de este archivo será la siguiente:
 - Se registrará por el formato de compresión MP3 (*MPEG Audio Layer III*).

10.1.3 Fichero de vídeo de la presentación

- Nombre: *video.mp4*
- Descripción: el sistema almacenará el vídeo generado por la presentación, en concreto, las diapositivas que el *usuario* este mostrando con ayuda del proyector, pero siendo estas grabadas directamente desde la fuente (desde el propio ordenador donde se esta ejecutando la presentación). El nombre de este archivo va a ser generado por el sistema informático de forma aleatoria, en caso de que no se especifique un montaje *offline*. Por otra parte, si el usuario especificase un montaje *offline*, este, le deberá indicar un nombre a este archivo, para posteriormente poder hacer referencia a el.
- Estructura: la estructura de este archivo será la siguiente:
 - Se registrará por el formato de compresión de vídeo *H.264*[14].

10.1.4 Fichero de vídeo procesado

- Nombre: *aleatorio.mkv*
- Descripción: el sistema almacenará el vídeo procesado, es decir, el vídeo que contiene las diapositivas de la presentación, con las coordenadas del puntero láser representadas en él. El nombre de este archivo va a ser generado por el sistema informático de forma aleatoria, ya que este archivo será completamente inexistente de cara al usuario.
- Estructura: la estructura de este archivo será la siguiente:
 - Se registrará por el formato de compresión de vídeo *H.264*[14].

10.1.5 Fichero de vídeo final

- Nombre: *final.mp4*
- Descripción: el sistema almacenará el vídeo final, es decir, el vídeo que contiene las diapositivas de la presentación, con las coordenadas del puntero láser representadas en él y el audio generado durante la presentación. El nombre de este archivo será especificado por el usuario, ya que es el archivo que contendrá el resultado final.
- Estructura: la estructura de este archivo será la siguiente:
 - Se registrará por el formato de compresión de vídeo *H.264*[14].
 - Como formato de compresión de audio se utilizará el formato *MP3* (*MPEG Audio Layer III*).

10.2 Diseño procedimental

En este apartado se van a detallar las clases que se tendrán que utilizar e implementar. Estas se van a extraer tanto de la especificación de requisitos como de los diagramas de secuencia o de la definición inicial que se hizo del problema.

En cada una de las clases, que se van a describir a continuación, se distinguirán principalmente los siguientes apartados:

- Nombre de la clase: nombre que se le va a asignar a la clase para su identificación. Este nombre, deberá ser descriptivo, de tal manera que proporcione una idea sobre el funcionamiento general del elemento al cual pertenece.
- Descripción general de la clase: se explicará el objetivo u objetivos de una forma breve y concisa, además de cualquier dato e información de interés sobre esta.
- Variables miembro de la clase: se analizarán todos los atributos que definan la clase y necesarios para almacenar la información necesaria

- *__track_model*: objeto encargado de realizar el *tracking* del puntero láser.
- *__grabacion_model*: objeto encargado de realizar la grabación de las diapositivas, el sonido y de la realización del montaje.
- *__camara_id*: número entero que contendrá el identificador de la cámara indicado por el usuario.
- *__sonido_id*: número entero que contendrá el identificador del dispositivo de captura de sonido indicado por el usuario.
- *__ruta_archivo*: cadena que contendrá la ubicación del archivo de captura indicada por el usuario.
- *__dir_salida*: cadena que contendrá la ubicación del directorio donde se guardará el archivo final correspondiente al montaje. Esta es indicada por el usuario.
- *__dir_mo*: cadena que contendrá la ubicación del directorio donde se guardarán / cargarán los archivos para la realización del montaje *offline*. Esta es indicada por el usuario.
- *__archivo_salida*: cadena que contendrá el nombre del archivo donde se guardará el vídeo final correspondiente al montaje. Esta es indicada por el usuario.
- *__arch_media_mo*: cadena que contendrá el nombre de los archivos correspondientes a la grabación de las diapositivas y el audio, para la realización del montaje *offline*. Esta es indicada por el usuario.
- *__arch_cap_mo*: cadena que contendrá el nombre del archivo que contiene la captura de las coordenadas del puntero láser. Este será necesario para la realización de un montaje *offline*. El nombre del archivo es indicado por el usuario.
- *__logica*: objeto encargado de enlazar la parte lógica del programa, con la interfaz de usuario. Su función principal será mostrar el progreso a la hora de realizar el montaje.
- *__formato_pantalla*: valor entero que indicará el formato de las diapositivas. Su valor podrá ser “0” para indicar una relación de

aspecto de 4:3, “1” para 16:9 y “-1” para una detección automática de la relación de aspecto³.

- *__ext_ancho*: valor entero que contendrá el ancho de la pantalla principal, indicado por el usuario. Este valor se utilizará cuando se use la pantalla extendida.
- *__ext_alto*: valor entero que contendrá el alto de la pantalla principal, indicado por el usuario. Este valor se utilizará cuando se use la pantalla extendida.

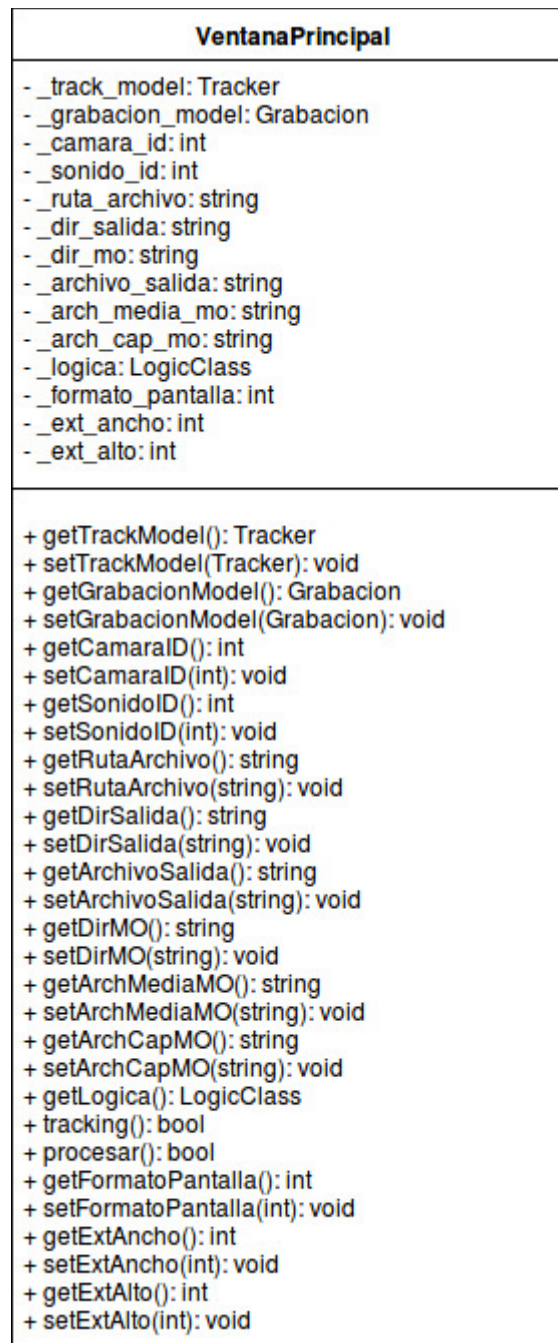
■ Métodos:

- *VentanaPrincipal*: Constructor de la clase, el cual se encargará de inicializar las variables de la clase.
- *getTrackModel*: obtendrá el objeto encargado de realizar el *tracking* del puntero láser.
- *setTrackModel*: establecerá el objeto encargado de realizar el *tracking* del puntero láser.
- *getGrabacionModel*: obtendrá el objeto encargado de realizar la grabación de las diapositivas, el sonido y de la realización del montaje.
- *setGrabacionModel*: establecerá el objeto encargado de realizar la grabación de las diapositivas, el sonido y de la realización del montaje.
- *getCamaraID*: obtendrá el valor entero que contiene el identificador de la cámara.
- *setCamaraID*: establecerá el valor entero que contiene el identificador de la cámara.
- *getSonidoID*: obtendrá el valor entero que contiene el identificador del dispositivo de captura de sonido.

³ La relación de aspecto [6] de una imagen, es la proporción entre su ancho y su altura. Se calcula dividiendo el ancho por la altura de la imagen visible en pantalla, y se expresa normalmente como X:Y

- *setSonidoID*: establecerá el valor entero que contiene el identificador del dispositivo de captura de sonido.
- *getRutaArchivo*: obtendrá la cadena que contiene la ubicación del archivo de captura.
- *setRutaArchivo*: establecerá la cadena que contiene la ubicación del archivo de captura.
- *getDirSalida*: obtendrá la cadena que contiene la ubicación del directorio donde se guardará el archivo final correspondiente al montaje.
- *setDirSalida*: establecerá la cadena que contiene la ubicación del directorio donde se guardará el archivo final correspondiente al montaje.
- *getArchivoSalida*: obtendrá la cadena que contiene el nombre del archivo donde se guardará el vídeo final correspondiente al montaje.
- *setArchivoSalida*: establecerá la cadena que contiene el nombre del archivo donde se guardará el vídeo final correspondiente al montaje.
- *getDirMO*: obtendrá la cadena que contiene la ubicación del directorio donde se guardarán / cargarán los archivos para la realización del montaje *offline*.
- *setDirMO*: establecerá la cadena que contiene la ubicación del directorio donde se guardarán / cargarán los archivos para la realización del montaje *offline*.
- *getArchivoMediaMo*: obtendrá la cadena que contiene el nombre de los archivos correspondientes a la grabación de las diapositivas y el audio, para la realización del montaje *offline*.
- *setArchivoMediaMo*: establecerá la cadena que contiene el nombre de los archivos correspondientes a la grabación de las diapositivas y el audio, para la realización del montaje *offline*.
- *getArchCapMo*: obtendrá la cadena que contiene el nombre del archivo que contiene la captura de las coordenadas del puntero láser. Este será necesario para la realización de un montaje *offline*.

- *setArchCapMo*: establecerá la cadena que contiene el nombre del archivo que contiene la captura de las coordenadas del puntero láser. Este será necesario para la realización de un montaje *offline*.
- *getLogica*: obtendrá el objeto encargado de enlazar la parte lógica del programa, con la interfaz de usuario.
- *tracking*: este método se utilizará para realizar la captura del puntero láser y la grabación del audio y vídeo correspondiente a las diapositivas mostradas por el usuario en la presentación. Para ello realizará una llamada a los métodos de la clase *Tracker* y *Grabación*.
- *procesar*: este método se utilizará para realizar el montaje de la presentación, representará las coordenadas del puntero láser, en la grabación realizada de las diapositivas y unirá el audio y vídeo resultantes. Para ello realizará una llamada a los métodos de la clase *Tracker* y *Grabación*.
- *getFormatoPantalla*: obtendrá el valor entero que indicará el formato de las diapositivas.
- *setFormatoPantalla*: establecerá el valor entero que indicará el formato de las diapositivas.
- *getExtAncho*: obtendrá el valor entero que contendrá el ancho de la pantalla principal, indicado por el usuario.
- *setExtAncho*: establecerá el valor entero que contendrá el ancho de la pantalla principal, indicado por el usuario.
- *getExtAlto*: obtendrá el valor entero que contendrá el alto de la pantalla principal, indicado por el usuario.
- *setExtAlto*: establecerá el valor entero que contendrá el alto de la pantalla principal, indicado por el usuario.



*Figura 10.2: Clase
VentanaPrincipal*

En la *Figura 10.2* se puede observar el diagrama correspondiente a la clase *VentanaPrincipal*.

10.2.2 Capa lógica

En esta sección se va a detallar el conjunto de clases que forman parte del único componente existente en la capa “Lógica”.

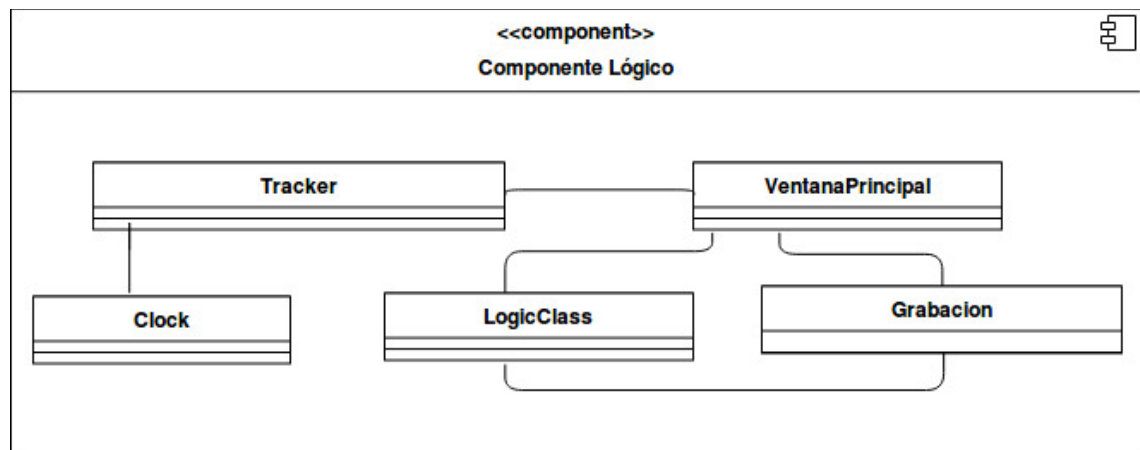


Figura 10.3: Componente correspondiente con la capa lógica

10.2.2.1 Clase tracker

En esta sección se va a detallar la clase encargada de realizar el seguimiento del puntero láser físico.

- Nombre: Tracker
- Descripción: Esta clase es la encargada de realizar el seguimiento del puntero láser y la obtención de sus correspondientes coordenadas.
- Variables miembro:
 - `__captura`: objeto de la clase `VideoCapture` de `OpenCV` que se encargará de obtener las imágenes de una cámara o un archivo de vídeo.
 - `__capturaAbierta`: valor lógico que indicará si se ha abierto el canal de captura de vídeo.

- *__bg_model*: puntero a un objeto de la clase *BackgroundSubtractor*, el cual se encargará de la extracción del fondo de la imagen para obtener el puntero láser.
 - *__img*: objeto de clase *Mat* de *OpenCV* que almacenará la imagen captura por la cámara.
 - *__fgimg*: objeto de clase *Mat* de *OpenCV* que almacenará la imagen correspondiente al fondo de la escena.
 - *__fgmask*: objeto de clase *Mat* de *OpenCV* que almacenará la imagen correspondiente a el puntero láser localizado.
 - *__posicion*: punto de dos dimensiones que indicará la posición del puntero láser físico.
 - *__vloc*: vector que almacenara un conjunto de estructuras *localizacion* (estructura definida por un punto de dos dimensiones y un número real) el cual corresponderá con la posición del puntero láser respecto al tiempo.
 - *__reloj*: objeto de la clase *Clock* que se encargará de realizar un seguimiento del tiempo transcurrido.
 - *__esArchivo*: valor lógico que indicará si la captura se quiere realizar desde una cámara o desde un archivo de vídeo.
 - *__region_de_interes*: objeto de la clase *Rect2d* de *OpenCV* el cual indicará la superficie de proyección seleccionada por el usuario.
 - *__resolucion*: punto de dos dimensiones que indicará la resolución correspondiente a la superficie de proyección seleccionada por el usuario.
- Métodos:
- *inicializar*: inicializará las variable de la clase.
 - *capturar*: este método se utilizará para capturar una imagen de la cámara o el archivo de captura.
 - *relase*: este método se utilizará para finalizar la captura y cerrar la cámara o el archivo.

- *extraerFondo*: este método se utilizará para extraer el fondo estático de la imagen para así poder detectar el puntero láser.
- *operacionesMorfologicas*: este método realizará una serie de operaciones morfológicas a la máscara obtenida después de la extracción del fondo, la cual contiene al puntero láser físico.
- *calcularPosicion*: calculará la posición de el puntero láser, dada la máscara con las operaciones morfológicas ya realizadas.
- *dibujarCirculo*: dibujará un círculo alrededor del puntero láser en la imagen original.
- *estaAbierta*: obtendrá el valor lógico que indicará si se ha abierto el canal de captura de vídeo.
- *getImagen*: obtendrá la imagen captada por la cámara o el archivo.
- *getFgimg*: obtendrá la imagen correspondiente al fondo de la escena.
- *getFgmask*: obtendrá la imagen correspondiente a el puntero láser localizado.
- *getPosicion*: obtendrá el punto de dos dimensiones que indicará la posición del puntero láser físico.
- *getLocalizacion*: obtendrá el vector que almacenara un conjunto de estructuras *localizacion*, el cual corresponderá con la posición del puntero láser respecto al tiempo.
- *getRegionInteres*: obtendrá el objeto de la clase Rect2d, el cual indicará la superficie de proyección seleccionada por el usuario.
- *calcularResolucion*: este método calculará la resolución correspondiente a la superficie de proyección seleccionada por el usuario.
- *getResolucion*: obtendrá el punto de dos dimensiones que indicará la resolución.
- *setResolucion*: establecerá el punto de dos dimensiones que indicará la resolución.

- *guardarArchivoLoc*: este método guardará en un archivo la localización del puntero láser respecto al tiempo.
- *cargarArchivoLoc*: este método cargará desde un archivo la localización del puntero láser respecto al tiempo.
- *getCaptura*: obtendrá el objeto de la clase *VideoCapture* de *OpenCV* que se encargará de obtener las imágenes de una cámara o un archivo de vídeo.
- *setCaptura*: establecerá el objeto de la clase *VideoCapture* de *OpenCV* que se encargará de obtener las imágenes de una cámara o un archivo de vídeo.
- *setEstaAbierta*: establecerá el valor lógico que indicará si se ha abierto el canal de captura de vídeo.
- *getBgModel*: obtendrá el puntero a un objeto de la clase *BackgroundSubtractor*, el cual se encargará de la extracción del fondo de la imagen para obtener el puntero láser.
- *setBgModel*: establecerá el puntero a un objeto de la clase *BackgroundSubtractor*, el cual se encargará de la extracción del fondo de la imagen para obtener el puntero láser.
- *setImagen*: establecerá la imagen captada por la cámara o el archivo.
- *setFgimg*: establecerá la imagen correspondiente al fondo de la escena.
- *setFgmask*: establecerá la imagen correspondiente a el puntero láser localizado.
- *setPosicion*: establecerá el punto de dos dimensiones que indicará la posición del puntero láser físico.
- *setLocalizacion*: establecerá el vector que almacenara un conjunto de estructuras *localizacion*, el cual corresponderá con la posición del puntero láser respecto al tiempo.

- *getReloj*: obtendrá el objeto de la clase *Clock* que se encargará de realizar un seguimiento del tiempo transcurrido.
- *setReloj*: establecerá el objeto de la clase *Clock* que se encargará de realizar un seguimiento del tiempo transcurrido.
- *esArchivo*: obtendrá el valor lógico que indicará si la captura se quiere realizar desde una cámara o desde un archivo de vídeo.
- *setEsArchivo*: establecerá el valor lógico que indicará si la captura se quiere realizar desde una cámara o desde un archivo de vídeo.
- *obtenerSuperficieProyeccion*: este método se encargará de pedirle al usuario que seleccione la superficie donde se están proyectando las diapositivas.
- *setRegionInteres*: establecerá el objeto de la clase *Rect2d*, el cual indicará la superficie de proyección seleccionada por el usuario.

En la *Figura 10.3* se puede observar el diagrama correspondiente a la clase *Tracker*.

Tracker
<ul style="list-style-type: none"> - _captura: VideoCapture - _capturaAbierta: bool - _bg_model: Ptr<BackgroundSubtractor> - _img: Mat - _fgimg: Mat - _fgmask: Mat - _posicion: Point2f - _vloc: vector <localizacion> - _reloj: Clock - _esArchivo: bool - _region_de_interes: Rect2d - _resolucion: Point2f
<ul style="list-style-type: none"> + inicializar(string): bool + inicializar(int): bool + capturar(): bool + release(): void + extraerFondo(): void + operacionesMorfologicas(int): void + calcularPosicion(Point2f): Point2f + dibujarCirculo(int, int): Mat + estaAbierta(): bool - setEstaAbierta(bool): void + getImagen(): Mat - setImagen(Mat): void + getFgimg(): Mat - setFgimg(Mat): void + getFgmask(): Mat - setFgmask(Mat): void + getPosicion(): Point2f - setPosicion(Point2f): void + getLocalizacion(): vector<localizacion> - setLocalizacion(vector<localizacion>): void + getRegionInteres(): Rect2d - setRegionInteres(Rect2d): void + calcularResolucion(): Point2f + getResolucion(): Point2f + setResolucion(Point2f): void + guardarArchivoLoc(string): bool + cargarArchivoLoc(string): bool - iniciarReloj(): void - getCaptura(): VideoCapture - setCaptura(VideoCapture): void - getBgModel(): Ptr<BackgroundSubtractor> - setBgModel(Ptr<BackgroundSubtractor>): void - getReloj(): Clock - setReloj(Clock): void - esArchivo(): bool - setEsArchivo(bool): void - obtenerSuperficieProyeccion: bool

*Figura 10.4: Clase
Tracker*

10.2.2.2 Clase Grabacion

En esta sección se va a detallar la clase encargada de realizar la grabación de las diapositivas y el audio, y del montaje.

- Nombre: Grabacion
- Descripción: Esta clase es la encargada de realizar la grabación de las dispositivas en formato digital y del audio generado durante la presentación. También se encarga del montaje de esta, es decir, teniendo las coordenadas del puntero láser, representa estas en el archivo de vídeo que contiene las diapositivas grabadas y finalmente unir el resultado con el archivo de audio.
- Variables miembro:
 - `__pid_audio`: valor de tipo `pid_t` que indicará el `pid`⁴ del proceso correspondiente a la aplicación de captura de audio.
 - `__pid_video`: valor de tipo `pid_t` que indicará el `pid` del proceso correspondiente a la aplicación de grabación del vídeo correspondiente a las diapositivas.
 - `__ancho`: valor entero que contendrá el ancho de la pantalla del ordenador.
 - `__alto`: valor entero que contendrá el alto de la pantalla del ordenador.
 - `__ruta_tmp`: cadena que indicará la ubicación de los archivos generados durante el proceso de grabación y procesado.
 - `__iniciada`: valor lógico que indicará si la grabación ha sido iniciada o no.
 - `__hw_sonido`: valor entero que contendrá el identificador del dispositivo de entrada de sonido.

⁴ *PID es una abreviatura de process ID, o sea, ID del proceso o bien identificador de procesos. El identificador de procesos es un número entero usado por el kernel de algunos sistemas operativos (como el de Unix o el de Windows NT) para identificar un proceso de forma unívoca[21].*

- *__formato_pantalla*: valor entero que indicará el formato de las diapositivas. Su valor podrá ser “0” para indicar una relación de aspecto de 4:3, “1” para 16:9 y “-1” para una detección automática de la relación de aspecto.

■ Métodos:

- *iniciar*: este método se encargará de iniciar la captura del audio y las diapositivas de la presentación (realizando una grabación del escritorio).
- *parar*: este método se encargará de parar la captura del audio y vídeo.
- *procesar*: este método se encargará de dibujar las coordenadas del puntero láser en las diapositivas grabadas, y posteriormente unirá el audio y el video generado en un archivo de vídeo.
- *getFormatoPantalla*: obtendrá valor entero que indicará el formato de las diapositivas.
- *setFormatoPantalla*: asignará el valor entero que indicará el formato de las diapositivas.
- *getRuta*: obtendrá la cadena que indicará la ubicación de los archivos generados durante el proceso de grabación y procesado.
- *setRuta*: establecerá la cadena que indicará la ubicación de los archivos generados durante el proceso de grabación y procesado.
- *getPidAudio*: obtendrá el valor de tipo *pid_t* que indicará el *pid* del proceso correspondiente a la aplicación de captura de audio.
- *setPidAudio*: asignará el valor de tipo *pid_t* que indicará el *pid* del proceso correspondiente a la aplicación de captura de audio.
- *getPidVideo*: obtendrá el valor de tipo *pid_t* que indicará el *pid* del proceso correspondiente a la aplicación de grabación del vídeo correspondiente a las diapositivas.

- *setPidVideo*: asignará el valor de tipo *pid_t* que indicará el *pid* del proceso correspondiente a la aplicación de grabación del vídeo correspondiente a las diapositivas.
- *getAncho*: obtendrá el valor entero que contendrá el ancho de la pantalla del ordenador.
- *setAncho*: asignará el valor entero que contendrá el ancho de la pantalla del ordenador.
- *getAlto*: obtendrá el valor entero que contendrá el alto de la pantalla del ordenador.
- *setAlto*: establecerá el valor entero que contendrá el alto de la pantalla del ordenador.
- *getIniciada*: obtendrá el valor lógico que indicará si la grabación ha sido iniciada o no.
- *setIniciada*: asignará el valor lógico que indicará si la grabación ha sido iniciada o no.
- *getHwSonido*: obtendrá el valor entero que contendrá el identificador del dispositivo de entrada de sonido.
- *setHwSonido*: asignará el valor entero que contendrá el identificador del dispositivo de entrada de sonido.
- *existeArchivo*: este método se encargará de comprobar si existe un archivo dado.
- *combinar*: este método se encargará de unir el audio grabado con el vídeo correspondiente a la grabación de las diapositivas ya procesadas con el puntero láser representado en ellas.
- *nombreAleatorio*: este método genera un nombre aleatorio para los archivos que se usarán durante el proceso de grabación y procesado.

En la *Figura 10.4* se puede observar el diagrama correspondiente a la clase *Tracker*.



Figura 10.5: Clase Grabacion

10.2.2.3 Clase LogicClass

En esta sección se va a detallar la clase *LogicClass*.

- Nombre: LogicClass
- Descripción: Esta clase es la encargada de enlazar la parte lógica del programa, con la interfaz de usuario. Su función principal será mostrar el progreso a la hora de realizar el montaje.
- Variables miembro: no posee.
- Métodos:
 - *max*: obtendrá el valor máximo de la barra de progreso.
 - *min*: obtendrá el valor mínimo de la barra de progreso.

- *EmitProgress*: emitirá el porcentaje del progreso de montaje.
- *emitFrame*: emitirá el número de *frame* por el que va el proceso de montaje.
- *emitNframes*: emitirá el número total de *frames*.
- *signalProgress*: asignará el porcentaje del progreso de montaje.
- *signalFrame*: asignará el número de *frame* por el que va el proceso de montaje.
- *signalNframes*: establecerá el número total de *frames*.

En la *Figura 10.5* se puede observar el diagrama correspondiente a la clase *LogicClass*.

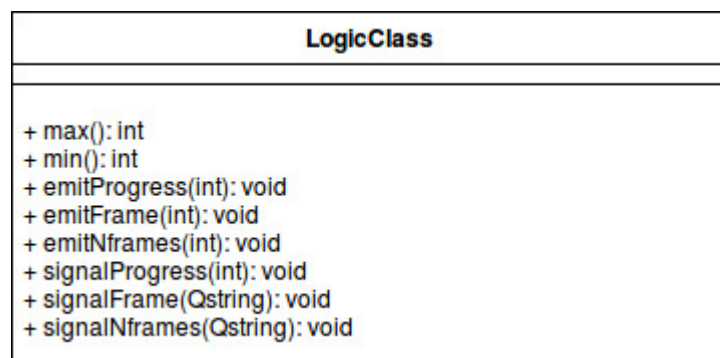


Figura 10.6: Clase LogicClass

10.2.2.4 Clase Clock

En esta sección se va a detallar la clase *Clock*.

- Nombre: *Clock*
- Descripción: esta clase se encarga de medir el tiempo transcurrido.
- Variables miembro:
 - *__start*: valor de tipo *timespec* que indicará el tiempo de inicio en microsegundos.

- `__stop`: valor de tipo *timespec* que indicará el tiempo de inicio en microsegundos, una vez parado el reloj.
- `__isStarted`: valor lógico que indicará si se ha iniciado el reloj.
- Métodos:
 - `stop`: asignará el tiempo transcurrido a la variable `__stop`.
 - `start`: asignará el tiempo a la variable `__start` e iniciará el cronómetro.
 - `restart`: reiniciará el cronómetro.
 - `isStarted`: obtendrá un valor lógico el cual indicará si se ha iniciado el cronómetro.
 - `elapsed`: obtendrá un valor de tipo *uint64_t* el cual indicará el tiempo transcurrido desde que se inicio el cronómetro.

En la *Figura 10.6* se puede observar el diagrama correspondiente a la clase *LogicClass*.

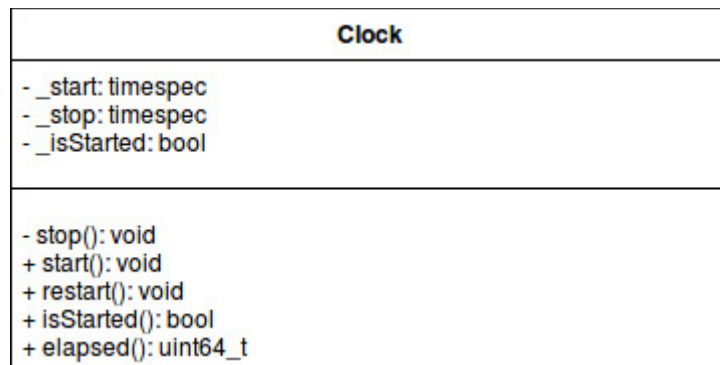


Figura 10.7: Clase Clock

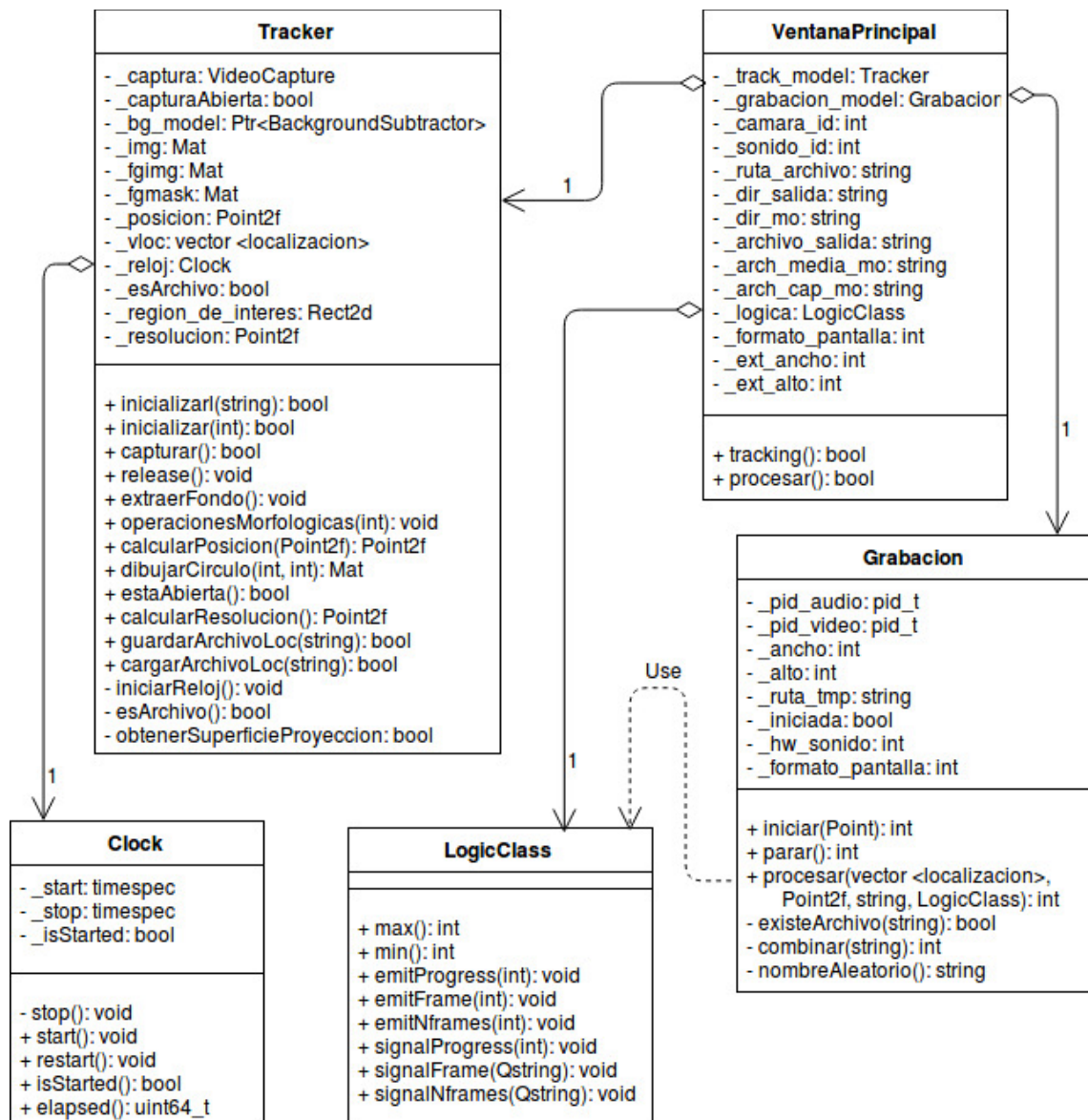


Figura 10.8: Diagrama de clases

10.3 Diseño de la Interfaz de Usuario

En este capítulo se va a describir el aspecto que tendrá de manera definitiva la interfaz de usuario de la aplicación, según las especificaciones de la misma que se han realizado previamente. El diseño de la interfaz se va a centrar en el diseño que mostrarán las principales pantallas y diálogos que se

desarrollarán. También se van a describir otros tipo de elementos, como el aspecto que van a presentar las diversas zonas y secciones en las que van a estar divididas dichas pantallas y diálogos, así como los distintos componentes contenidos en estas secciones y su funcionalidad.

A continuación se van a ir mostrando las diversas pantallas que se han desarrollado y las cuales cumplen con las especificaciones realizadas en el apartado de especificación de la interfaz. Para ello la aplicación va a constar de una única ventana, dividida en un conjunto de pestañas, estas se pasan a definir a continuación.

10.3.1 Pestaña captura

La función principal de esta pestaña es la de proporcionar al usuario la funcionalidad de realizar la captura de la presentación, tanto el puntero láser como el audio y el vídeo correspondiente a las diapositivas de esta.

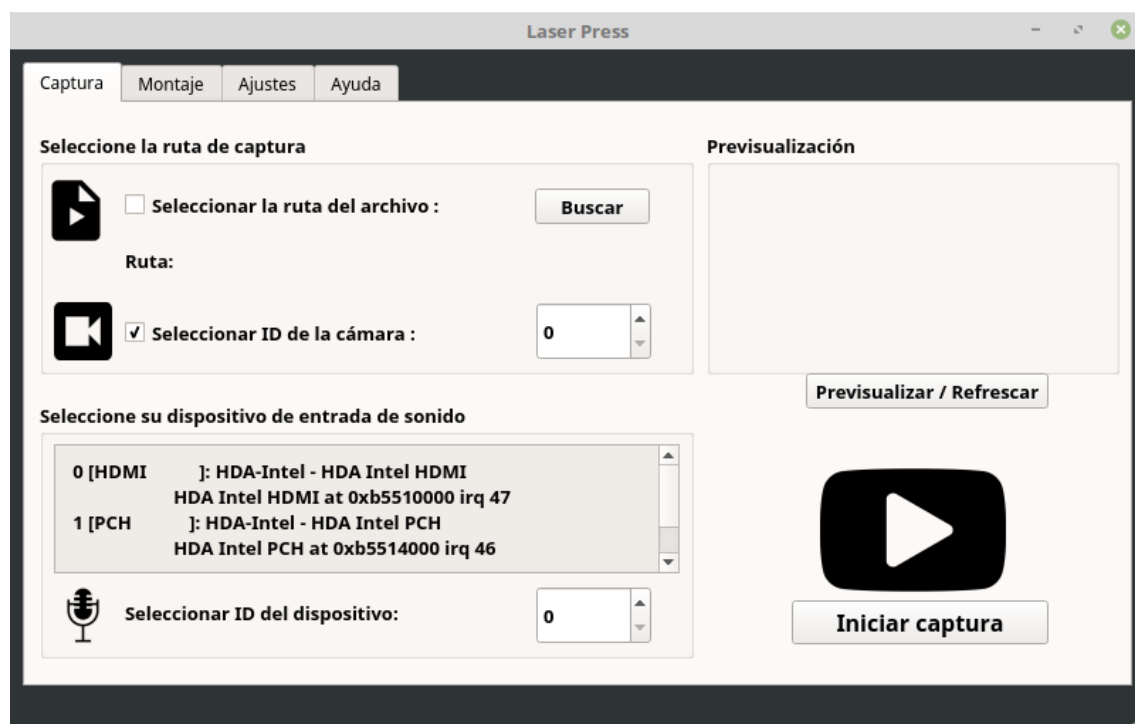


Figura 10.9: Pestaña captura

El diseño que se ha realizado para la pestaña *captura* es el que se muestra en la figura siguiente:

Como se puede observar en la *Figura 10.9*, la pestaña captura de la aplicación dispone de cuatro partes principales:

- “Seleccione la ruta de captura”: en esta sección se pueden distinguir dos opciones a elegir:
 - “Seleccionar la ruta del archivo”: con esta opción se puede seleccionar un archivo desde el que capturar la presentación (puntero láser físico y diapositivas). El botón situado a su derecha (“Buscar”), llevará a una nueva ventana en la cual se podrá seleccionar dicho archivo, tal como se observa en la *Figura 10.10*.

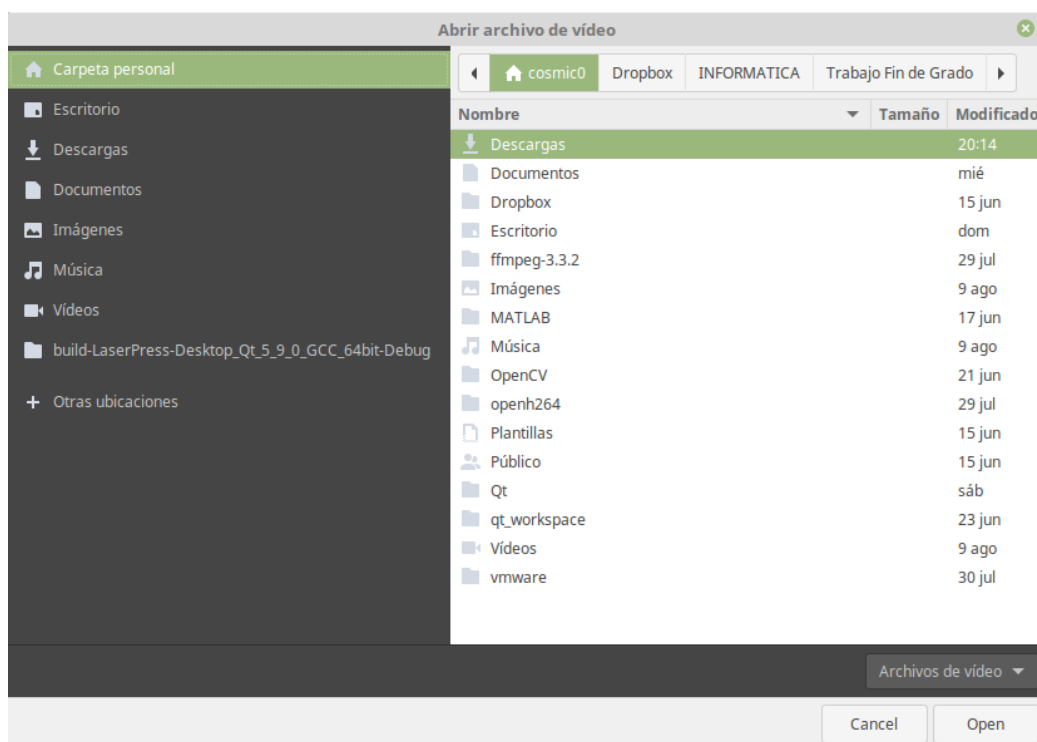


Figura 10.10: Ventana buscar archivo

- “Seleccionar ID de la cámara”: con esta opción se puede seleccionar el identificador de un dispositivo de captación de imágenes que esté conectado al ordenador. Generalmente el dispositivo por defecto

será el “0”, posteriormente conforme se añadan nuevos dispositivos, el número del identificador será el siguiente a el anteriormente conectado. Para ello se pone a disposición del usuario un elemento gráfico mediante el cual se podrá indicar el identificador.

- “Previsualización”: en esta sección, se podrá realizar una previsualización de lo que está captando la cámara o el archivo. De esta forma se podrá saber si la cámara seleccionada es la que se quiere usar. Para ello, se pone a disposición del usuario un elemento gráfico mediante el cual se podrá realizar la previsualización del dispositivo de captura.
- “Seleccione su dispositivo de entrada de sonido”: con esta opción se puede seleccionar el identificador de un dispositivo de captación de sonido que esté conectado al ordenador. Para ello se muestra en un marco, el conjunto de dispositivos de sonido conectados al ordenador con su identificador correspondiente. También se pone a disposición del usuario un elemento gráfico mediante el cuál podrá seleccionar el identificador.
- “Iniciar captura”: esta sección esta formada por un único elemento gráfico (un botón) el cual servirá como disparador de inicio de la captura. Concretamente se procederá a captar el puntero láser e iniciar la grabación de las diapositivas y el audio de la presentación.

10.3.2 Pestaña montaje

La función principal de esta pestaña es la de proporcionar al usuario la funcionalidad de realizar el montaje de la captura previamente realizada.

El diseño que se ha realizado para pestaña *montaje* es el que se muestra en la figura siguiente:

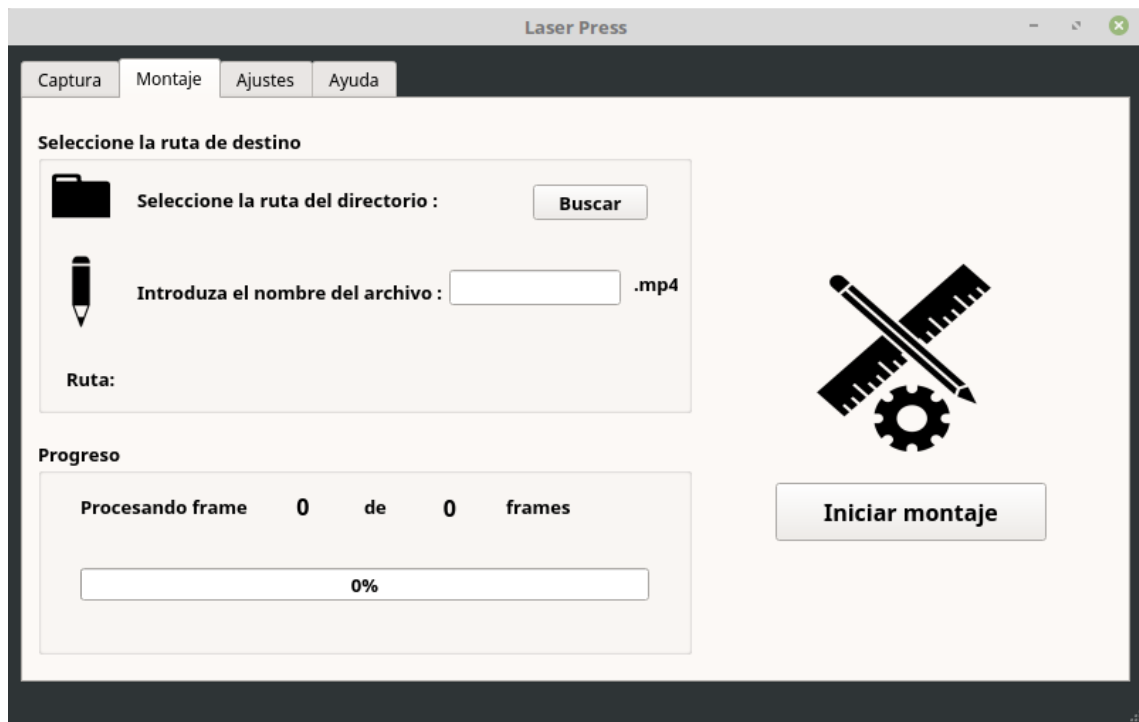


Figura 10.11: Pestaña montaje

Como se puede observar en la *Figura 10.11*, la pestaña *montaje* de la aplicación dispone de tres parte principales:

- “Seleccione la ruta de destino”: en esta sección se pueden distinguir dos partes principales:
 - “Seleccione la ruta del directorio”: con esta opción se puede seleccionar el directorio donde se guardará el resultado final del montaje. El botón situado a su derecha (“Buscar”), llevará a una nueva ventana en la cual se podrá seleccionar el directorio.
 - “Introduzca el nombre del archivo”: en este apartado se pone a disposición del usuario un campo de introducción de texto, para que introduzca el nombre que tendrá el archivo del montaje final, a este nombre se le introducirá la extensión “.mp4”. Finalmente la ruta

completa del archivo se mostrará en la parte inferior, concretamente en la zona titulada como “Ruta”.

- **“Progreso”**: esta sección sirve como apartado informativo, para el usuario, del progreso de montaje. Para ello se pone a disposición del usuario el número de *frames* que se llevan procesados y el total de ellos, además de una barra de progreso que indicará el porcentaje del montaje que lleva completado.
- **“Iniciar montaje”**: esta sección esta formada por un único elemento gráfico (un botón) el cual servirá como disparador de inicio del montaje. Concretamente se procederá a representar las coordenadas del puntero láser en la grabación realizada de las diapositivas, y la unión de esta con el audio generado durante la presentación.

10.3.3 Pestaña ajustes

La función principal de esta pestaña es la de proporcionar al usuario la posibilidad de poder realizar diversos ajustes tanto para la etapa de captura, como la de montaje.

El diseño que se ha realizado para la pestaña *ajustes* es el que se muestra en la *Figura 10.12*. Como se puede observar, la pestaña *información* de la aplicación dispone de tres partes principales:

- **“Seleccionar la relación de aspecto de la presentación”**: en esta sección, se pueden distinguir tres opciones a elegir:
 - **“Detección automática”**: con esta opción se puede indicar al programa que detecte la relación de aspecto de la superficie de proyección (donde se proyecta la presentación) de forma automática.
 - **“Relación de aspecto 4:3”**: con esta opción se establecerá la relación de aspecto de la superficie de proyección a “4:3”, de esta forma si se está proyectando con una relación de aspecto de “16:9” y la

presentación tiene un formato de “4:3” se podrán obtener las coordenadas de forma fidedigna.

- “Relación de aspecto 16:9”: con esta opción se establecerá la relación de aspecto de la superficie de proyección a “16:9”, de esta forma si se está proyectando con una relación de aspecto de “4:3” y la presentación tiene un formato de “16:9” se podrán obtener las coordenadas de forma fidedigna.
- “Usar pantalla extendida”: con esta opción el usuario puede indicarle al programa que va a usar la pantalla de su ordenador de forma extendida, para ello deberá indicarle el ancho y alto de la pantalla principal. Por lo que, se ponen a disposición del usuario dos elementos gráficos (campos de introducción de datos numéricos) para que introduzca el ancho y alto de la pantalla.
- “Ajustes para realización de un montaje offline”: en esta sección se pueden introducir los datos necesarios para la realización de un montaje *offline*. Se encuentran definidos dos apartados:
 - “Ajustes de captura”: aquí se podrá seleccionar donde se van a guardar los archivos de la captura, para poder realizar el montaje posteriormente. Para ello el usuario podrá seleccionar a través del botón “Buscar” el directorio donde serán guardados dichos ficheros. También se ponen a disposición del usuario, dos campos de introducción de texto, para que pueda introducir el nombre de los archivos de audio y vídeo, correspondientes con el audio generado durante la presentación y el vídeo de las diapositivas, y el nombre del archivo de captura.
 - “Ajustes de montaje”: aquí se podrá seleccionar de donde se van a cargar los archivos de la captura, para poder realizar el montaje. Para ello el usuario podrá seleccionar a través del botón “Buscar” el directorio donde serán guardados dichos ficheros. También se pone a disposición del usuario, un campo de introducción de texto, para que pueda introducir el nombre de los archivos de audio y

vídeo, correspondientes con el audio generado durante la presentación y el vídeo de las diapositivas. También se incluye un botón (“Buscar”) para que el usuario pueda seleccionar el archivo de captura.

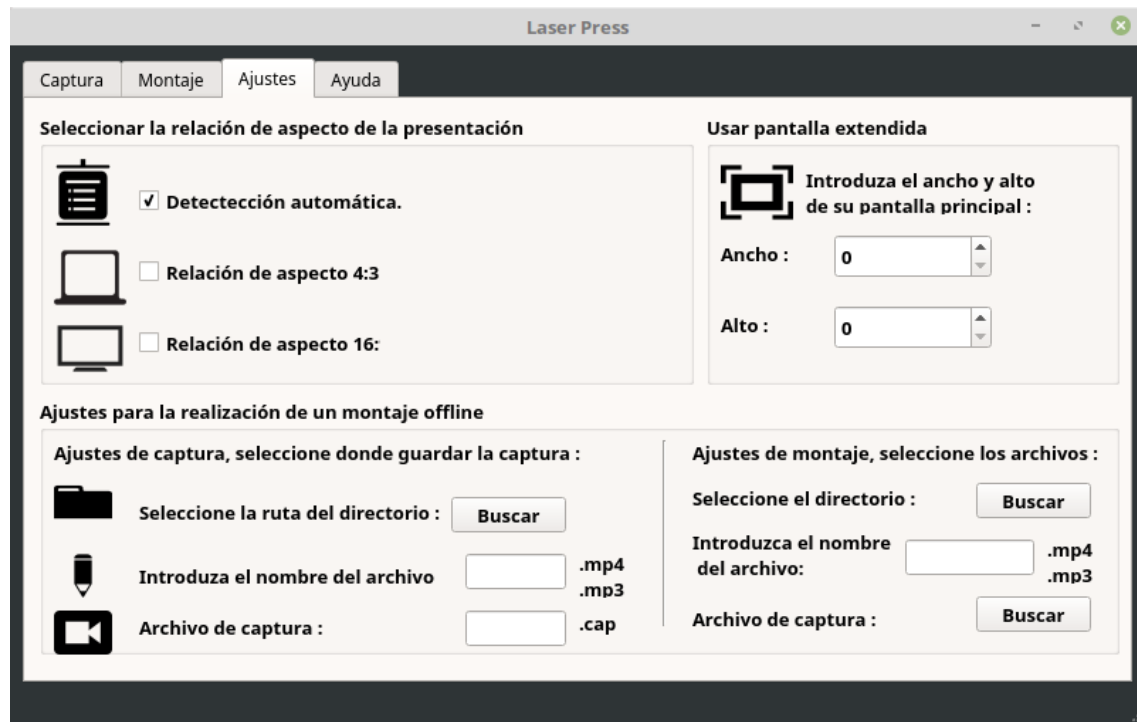


Figura 10.12: Pestaña ajustes

10.3.4 Pestaña información

En esta pestaña, no existirá ningún tipo de interacción por parte del usuario. En ella se muestra información sobre el programa y el autor del sistema informático.

El diseño que se ha realizado para la pestaña *información* es el que se muestra en la *Figura 10.13*.

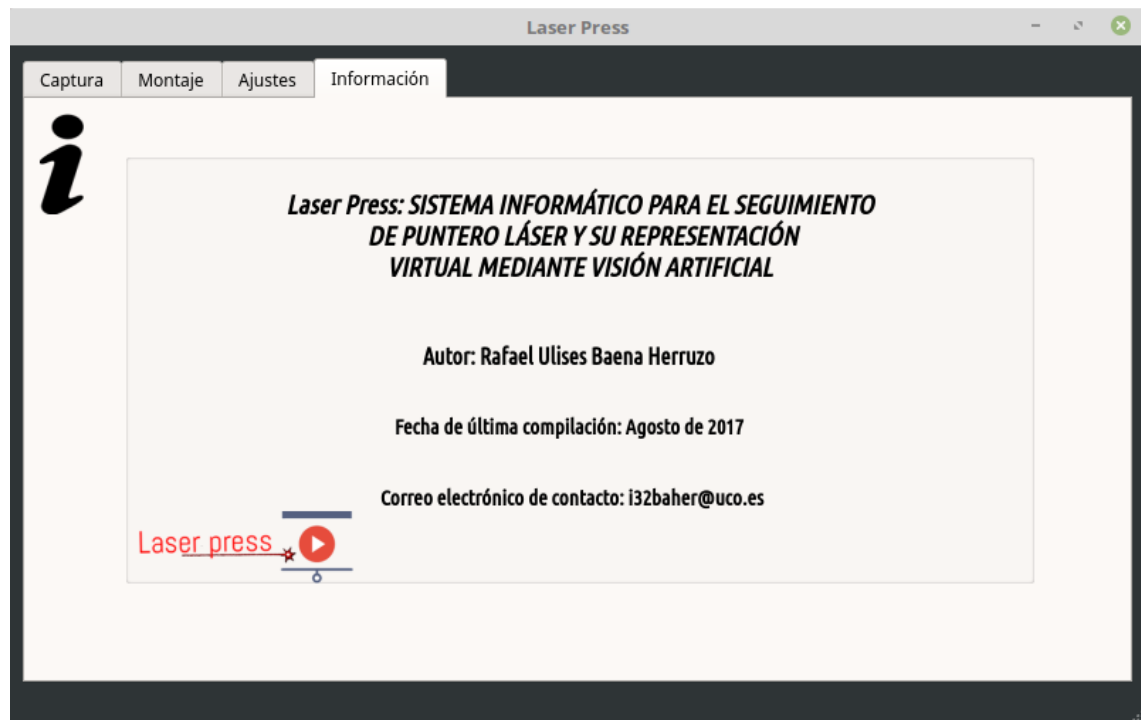


Figura 10.13: Pestaña información

Capítulo 11. Pruebas

En este capítulo se van a realizar las comprobaciones necesarias a la aplicación, para comprobar que esta cumple con el propósito por la que fue diseñada. Para ello se van a llevar a cabo dos técnicas que comprobarán que toda la funcionalidad de la aplicación proporciona la salida esperada. La primera técnica denominada Pruebas de Casos de Uso, consistirá en la realización de pruebas para cada uno de los casos de uso descritos en el capítulo 8, y la segunda técnica se denomina Pruebas de Escenarios de la Aplicación.

11.1 Pruebas de Casos de Uso

Para realizar las pruebas de casos de uso, se definirán casos de prueba para comprobar que se cumplen las especificaciones que se detallan en cada caso de uso.

11.1.1 Caso de prueba. Seleccionar medios de captura de vídeo

Nombre	Seleccionar medios de captura de vídeo
Condiciones:	Debe existir el archivo de vídeo para realizar la captura, o tener conectada una cámara al ordenador.
Salida esperada	Se espera que la aplicación permita al

	usuario poder realizar posteriormente la captura de manera correcta.
Salida obtenida	Se muestra la ruta del archivo seleccionado o el identificador de la cámara seleccionada.
Resultado	Correcto

Tabla 11.1: Caso de prueba: Seleccionar medios de captura de vídeo

11.1.2 Caso de prueba. Previsualizar medios de captura de vídeo

Nombre	Previsualizar medios de captura de vídeo
Condiciones:	El usuario debe haber seleccionado un medio de captura de vídeo previamente.
Salida esperada	Se espera que la aplicación muestre una imagen captura a través del dispositivo de captura seleccionado.
Salida obtenida	Se muestra la imagen capturada a través del dispositivo de captura seleccionado.
Resultado	Correcto

Tabla 11.2: Caso de prueba: Previsualizar medios de captura de vídeo

11.1.3 Caso de prueba. Seleccionar dispositivo de captura de audio

Nombre	Seleccionar dispositivo de captura de audio
Condiciones:	Debe haber un dispositivo de captura de audio conectado al ordenador.
Salida esperada	Se espera que la aplicación permita al usuario seleccionar el identificador del dispositivo de captura de audio, para posteriormente poder realizar la captura de manera correcta.
Salida obtenida	Se muestra el identificador del dispositivo de captura de audio seleccionado.
Resultado	Correcto

Tabla 11.3: Caso de prueba: Seleccionar dispositivo de captura de audio

11.1.4 Caso de prueba. Iniciar captura

Nombre	Iniciar captura
Condiciones:	El usuario debe haber seleccionado un dispositivo de captura de vídeo y de audio, los cuales estén disponibles.
Salida esperada	Se espera que la aplicación muestre una ventana en la cual el usuario pueda seleccionar la superficie de proyección.
Salida obtenida	Se muestra la ventana de selección de la superficie de proyección.
Resultado	Correcto

Tabla 11.4: Caso de prueba: Iniciar captura

11.1.5 Caso de prueba. Seleccionar superficie de proyección

Nombre	Seleccionar superficie de proyección
Condiciones:	El usuario debe haber iniciado la captura de la presentación de forma correcta.
Salida esperada	Se espera que la aplicación permita seleccionar la superficie de proyección para posteriormente mostrar como captura el puntero láser.
Salida obtenida	Se inicia una nueva ventana en la cual se ve como se realiza el seguimiento del puntero láser.
Resultado	Correcto

Tabla 11.5: Caso de prueba: Seleccionar superficie de proyección

11.1.6 Caso de prueba. Seleccionar ruta de destino

Nombre	Seleccionar ruta de destino
Condiciones:	No debe existir el archivo seleccionado.
Salida esperada	Se espera que la aplicación permita al usuario poder realizar posteriormente el montaje, guardando el resultado de este, en el archivo seleccionado.
Salida obtenida	Se muestra la ruta del archivo seleccionado.
Resultado	Correcto

Tabla 11.6: Caso de prueba: Seleccionar ruta de destino

11.1.7 Caso de prueba. Iniciar montaje

Nombre	Iniciar montaje
Condiciones:	El usuario debe haber seleccionado la ruta del archivo donde se guardará el montaje final.
Salida esperada	Se espera que la aplicación comience el montaje y muestre el progreso de este, y finalmente un mensaje avisando de que el montaje se ha realizado de manera correcta.
Salida obtenida	Se inicia el montaje y se muestra el progreso, hasta su finalización, mostrando un mensaje que indica que el montaje se ha realizado con éxito.
Resultado	Correcto

Tabla 11.7: Caso de prueba: Iniciar montaje

11.1.8 Caso de prueba. Seleccionar relación de aspecto

Nombre	Seleccionar relación de aspecto
Condiciones:	El usuario, no debe haber realizado el montaje.
Salida esperada	Se espera que la aplicación permita al usuario poder seleccionar la relación de aspecto de las diapositivas de la presentación.

Salida obtenida	El sistema permite la selección de la relación de aspecto. Posteriormente el montaje se realiza correctamente, habiendo seleccionado una relación de aspecto concreta, o dejando la detección automática de esta.
Resultado	Correcto

Tabla 11.8: Caso de prueba: Seleccionar relación de aspecto

11.1.9 Caso de prueba. Usar pantalla extendida

Nombre	Usar pantalla extendida
Condiciones:	El usuario, no debe haber iniciado la captura ni el montaje. Además debe tener la pantalla de su ordenador, conectada al proyector utilizando el formato de pantalla extendida.
Salida esperada	Se espera que la aplicación permita al usuario poder seleccionar la resolución de su pantalla principal para poder usar la pantalla extendida del ordenador.
Salida obtenida	El sistema permite la selección de la resolución de la pantalla principal de su ordenador. Posteriormente al realizar el montaje, el vídeo final solo muestra las diapositivas que se están proyectando en la pantalla extendida.
Resultado	Correcto

Tabla 11.9: Caso de prueba: Usar pantalla extendida

11.1.10 Caso de prueba. Montaje offline - guardar captura

Nombre	Montaje offline – guardar captura
Condiciones:	El usuario, no debe haber iniciado la captura ni el montaje.
Salida esperada	Se espera que la aplicación permita al usuario poder seleccionar la ruta donde se guardarán los archivos correspondientes a la captura.
Salida obtenida	El sistema permite seleccionar donde se van a localizar los archivos de la captura. Posteriormente al iniciar la captura, los archivos se guardan correctamente.
Resultado	Correcto

Tabla 11.10: Caso de prueba: Montaje offline – guardar captura

11.1.11 Caso de prueba. Montaje offline - cargar captura

Nombre	Montaje offline – cargar captura
Condiciones:	El usuario, no debe haber iniciado el montaje. Además, los archivos a seleccionar deben existir.
Salida esperada	Se espera que la aplicación permita al usuario poder seleccionar los archivos correspondientes a la captura.
Salida obtenida	El sistema permite seleccionar los

	archivos de la captura. Posteriormente al iniciar el montaje, los archivos se cargan correctamente.
Resultado	Correcto

Tabla 11.11: Caso de prueba: Montaje offline – cargar captura

11.2 Pruebas de Escenarios de la Aplicación

Se van a implementar una serie de casos de prueba, los cuales verificarán la interacción que realiza el usuario con los componentes que implementan cada caso de uso.

A continuación se describirán los distintos casos de prueba que han sido realizados para cada uno de los casos de uso y el resultado obtenido.

11.2.1 Caso de prueba. Seleccionar medios de captura de vídeo

Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario selecciona como dispositivo de captura un archivo, pero no indica el archivo.	Al iniciar la captura, el sistema informa mediante un mensaje de error que no se pudo abrir el archivo de vídeo.	Completado.	✓
2	El usuario selecciona como dispositivo de captura una cámara conectada al ordenador, pero el identificador es incorrecto.	Al iniciar la captura, el sistema informa mediante un mensaje de error que no se pudo abrir la cámara.	Completado.	✓

Tabla 11.12: Pruebas de Escenarios de la Aplicación: Seleccionar medios de captura de vídeo

11.2.2 Caso de prueba. Previsualizar medios de captura de vídeo

Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario no ha indicado el archivo o dispositivo de captura, o este no existe.	Al pulsar el botón previsualizar el sistema informa mediante un mensaje de error que no se pudo abrir la ruta de captura seleccionada.	Completado.	✓

Tabla 11.13: Pruebas de Escenarios de la Aplicación: Previsualizar medios de captura de vídeo

11.2.3 Caso de prueba. Seleccionar dispositivo de captura de audio

Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario al seleccionar un dispositivo de captura de audio, selecciona un identificador incorrecto.	Al finalizar la captura, el sistema informa mediante un mensaje de error que no se ha podido grabar el audio.	Completado.	✓

Tabla 11.14: Pruebas de Escenarios de la Aplicación: Seleccionar dispositivo de captura de audio

11.2.4 Caso de prueba. Seleccionar superficie de proyección


Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario no selecciona la superficie de proyección y pulsa la tecla “ESC”.	El sistema no puede recuperarse de este error y el programa finaliza de forma inesperada. La cámara se queda atrapada.	Completado.	

Tabla 11.15: Pruebas de Escenarios de la Aplicación: Seleccionar superficie de proyección

11.2.5 Caso de prueba. Seleccionar ruta de destino


Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario no indica ni directorio ni nombre del archivo, e inicia el montaje.	El sistema guarda el montaje en un archivo en la carpeta temporal del sistema.	Completado.	

Tabla 11.16: Pruebas de Escenarios de la Aplicación: Seleccionar ruta de destino

11.2.6 Caso de prueba. Iniciar montaje


Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario inicia el montaje sin haber realizado previamente la captura o indicado los archivos para un montaje offline.	Al iniciar el montaje, el sistema informa mediante un mensaje de error que no se pudo abrir el archivo de vídeo.	Completado.	

Tabla 11.17: Pruebas de Escenarios de la Aplicación: Iniciar montaje

11.2.7 Caso de prueba. Montaje offline – guardar captura

Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario selecciona solo el directorio donde guardar los archivos, o solo indica el nombre de estos.	El sistema se comporta como si no se hubiese seleccionado el montaje offline.	Completado.	✓

Tabla 11.18: Pruebas de Escenarios de la Aplicación: Montaje offline – guardar captura

11.2.8 Caso de prueba. Montaje offline – cargar captura

Caso	Descripción	Acciones esperadas	Resultado	Estado
1	El usuario selecciona solo el directorio donde cargar los archivos, o solo indica el nombre de estos.	El sistema se comporta como si no se hubiese seleccionado el montaje offline.	Completado.	✓

Tabla 11.19: Pruebas de Escenarios de la Aplicación: Montaje offline – cargar captura

Capítulo 12. Conclusiones

En este capítulo se van a exponer las conclusiones a las que se ha llegado una vez finalizadas las diferentes fases de desarrollo de la aplicación. Dichas conclusiones estarán relacionadas con los objetivos que se han planteado inicialmente y con los objetivos después de realizar la fase de pruebas. También se va a intentar llegar a una conclusión sobre las futuras mejoras que se podrían desarrollar en un futuro.

12.1 Conclusiones sobre los objetivos planteados

Se puede decir, que los objetivos que se plantearon en el capítulo 3 de esta manual, se han visto cumplidos de manera satisfactoria.

- El objetivo principal de proyecto consistía en desarrollar e implementar una aplicación que permitiese grabar una exposición pública con ayuda del proyector, representando virtualmente el seguimiento del puntero láser físico utilizado como apoyo de la presentación. Para ello, el sistema implementaría algoritmos de reconocimiento de objetos en movimiento. Por lo que se puede concluir, tal y como se ha demostrado en la fase de pruebas, que el objetivo principal ha sido llevado a cabo con éxito.
- Se ha desarrollado un algoritmo capaz de capturar el contenido de una videocámara y de localizar las coordenadas de un puntero láser situado en dichas capturas.

- Se ha desarrollado un método para procesar las coordenadas obtenidas y aplicárselas correctamente a las diapositivas capturadas desde el ordenador. También cabe destacar, que se ha implementado un método para poder adaptar la obtención de las coordenadas y su representación a distintas relaciones de aspecto de pantalla y a el uso de la pantalla extendida.
- Se ha desarrollado un método para permitir la realización de un montaje *offline* del vídeo final, permitiendo al usuario poder realizar el montaje en el momento que desee.
- Finalmente, el usuario dispone de una aplicación gráfica con una interfaz simple e intuitiva, la cual le permite realizar la captura de las diapositivas, el audio y el puntero láser físico, realizar el montaje de la captura previamente realizada e incluso un conjunto de herramientas para poder amoldar el programa a el uso del usuario, con una pestaña de ajustes completa, pero a la vez sencilla.

12.2 Conclusiones de la fase de pruebas

En la fase de pruebas se han ejecutado un conjunto de pruebas para comprobar que el sistema ofrece el total de funcionalidades esperadas y que no se producen errores, ni por parte de la interacción del usuario con la aplicación ni por parte de los métodos encargados de dar la funcionalidad al programa.

Cabe decir, que aunque la aplicación ha superado con éxito las pruebas diseñadas en dicha fase, no puede nunca asegurar al cien por cien que no existan errores, pues puede llegar a ocurrir un error en el que no se haya contemplado un caso donde pueda llegar a fallar. Por lo que, lo ideal sería poner la aplicación en manos de los usuarios (ajenos completamente al desarrollo) para que pudiesen probarla y detectar fallos en caso de tenerlos y así poder subsanarlos antes de obtener una versión definitiva.

12.3 Futuras mejoras

La aplicación cumple con todos los objetivos que se han ido planteando, pero se podrían añadir funcionalidades nuevas y mejorar el sistema de cara a futuras versiones.

Dichas funcionalidades podrían ser las siguientes:

- Desarrollar la aplicación para que pueda ser ejecutada en los principales sistemas (*Windows*, *MAC*).
- Poder grabar al conferenciante de la presentación y así poder alternar entre conferenciante y diapositivas.
- Agilizar el proceso de montaje, y en el futuro que se pueda realizar a la vez que se realiza la captura.
- Integrar plataformas de *streaming* de vídeo.
- Poder realizar la captura con más cámaras, captando distintas partes de interés de la presentación.
- Poder editar las partes capturadas de la presentación, indicando partes que no se quiera que estén presentes en el montaje.
- Detectar de manera automática la superficie de proyección, sin necesidad de que el usuario tenga que indicarla de forma manual.
- Utilizar las APIs existentes de grabación de audio y vídeo internamente en el programa.

Bibliografía

- [1] «C++», *Wikipedia, la enciclopedia libre*. 01-ago-2017.
- [2] «Atom (editor de textos)», *Wikipedia, la enciclopedia libre*. 06-ago-2017.
- [3] «Libav», *Wikipedia*. 22-jul-2017.
- [4] «OpenCV», *Wikipedia, la enciclopedia libre*. 14-mar-2017.
- [5] «Qt Creator», *Wikipedia, la enciclopedia libre*. 08-ago-2017.
- [6] «Relación de aspecto», *Wikipedia, la enciclopedia libre*. 11-jul-2017.
- [7] T. Fisher, «What is a Compressed File?», *Lifewire*. [En línea].
Disponible en: <https://www.lifewire.com/what-is-a-compressed-file-2625829>.
- [8] Rafael C. Gonzalez; Richard E. Woods; Digital Image Processing. 3^a
de. Londres. Pearson Education. 2010. 976 p. ISBN: 0-13-234563-3
- [9] «Libav documentation : avconv». [En línea]. Disponible en:
<https://libav.org/avconv.html>. [Accedido: 14-ago-2017].
- [10] «Writer | LibreOffice en español - el paquete de oficina por excelencia». [En línea]. Disponible en: <https://es.libreoffice.org/descubre/writer/>.
[Accedido: 14-ago-2017].
- [11] «Zotero | Home». [En línea]. Disponible en: <https://www.zotero.org/>.
[Accedido: 14-ago-2017].

- [12] «draw.io». [En línea]. Disponible en: <https://www.draw.io/>. [Accedido: 14-ago-2017].
- [13] «Main Page - Linux Mint». [En línea]. Disponible en: <https://www.linuxmint.com/>. [Accedido: 14-ago-2017].
- [14] «H.264/MPEG-4 AVC», *Wikipedia, la enciclopedia libre*. 22-jul-2017.
- [15] «FFmpeg», *Wikipedia, la enciclopedia libre*. 10-jun-2017.
- [16] C. Guindon, «Eclipse desktop & web IDEs». [En línea]. Disponible en: <https://eclipse.org/ide/>.
- [17] «GTK+», *Wikipedia, la enciclopedia libre*. 12-ago-2017.
- [18] «BoofCV». [En línea]. Disponible en: https://boofcv.org/index.php?title=Main_Page. [Accedido: 18-ago-2017].
- [19] opencv_contrib: Repository for OpenCV's extra modules. OpenCV, 2017.
- [20] Joseph Schmuller; Aprendiendo UML en 24 horas. 1^a de. México. Pearson Education Latinoamérica. 2000. 404 p. ISBN: 9789684444638
- [21] «Identificador de proceso», *Wikipedia, la enciclopedia libre*. 12-ago-2017.