Universidad Politécnica de Valencia

UNIVERSITAT POLITÈCNICA DE VALÈNCIA

Trabajo final de Máster

Desarrollo de un Escritorio Digital para la captura, transcripción y gestión multimodal e interactiva de documentos manuscritos

Autor:
Francisco Javier Vicente Parra

Tutor:
Francisco José Abad Cerdá

Trabajo final de
Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

realizado en el

Computer Graphics Group
Departamento de Sistemas Informáticos y Computación

8 de febrero de 2013

UNIVERSIDAD POLITÉCNICA DE VALENCIA

Resumen

Departamento de Sistemas Informáticos y Computación

Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital

Desarrollo de un Escritorio Digital para la captura, transcripción y gestión multimodal e interactiva de documentos manuscritos

por Francisco Javier Vicente Parra

El presente trabajo final de Máster consiste en el diseño e implementación de la interfaz humano-computador para el proyecto multidisciplinar titulado "Captura, transcripción y gestión multimodal e interactiva de documentos manuscritos".

Dicha interfaz tiene como objetivo facilitar al usuario la utilización del sistema, de forma que pueda realizar las tareas de dicho proyecto de una forma sencilla, intuitiva y eficaz. Para ello se ha diseñado un sistema básico de visualización y gestión de documentos, que permite interaccionar con las distintas partes del sistema, como son la captura de documentos, su procesamiento, edición, impresión y almacenamiento.

Agradecimientos

Agradecer en primer lugar a la Universidad Politécnica de Valencia, bajo el "Programa de Apoyo a la Investigación" (PAID-05-11), que ha dado lugar y financiado el proyecto multidisciplinar en el cual he contribuido con el proyecto desarrollado en esta tesina.

Agradezco a mi tutor, Paco, su ayuda y dedicación, pues pese a estar trabajando a miles de kilómetros de aquí, siempre ha respondido con rapidez y precisión todas las dudas y revisiones que han sido necesarias.

También al resto del equipo del proyecto multidisciplinar, comenzando por el responsable principal del proyecto, el Doctor Enrique Vidal Ruiz, y mis compañeros Verónica Romero Gómez y Daniel Martín-Albo Simón.

A mis padres, Joaquín y María; mi hermana Araceli; Alberto, Jorge, Miguel, Marian, Rubén, Virginia, Laura, Esmeralda, etc. entre otros muchos amigos (que si me pongo a nombrar ocuparía demasiado, que nadie se ofenda por no aparecer en la escueta lista de los seleccionados) y compañeros que siempre han estado detrás, apoyándome y animándome.

Sin olvidar a los compañeros de laboratorio del grupo de gráficos, que siempre han estado dispuestos a echar una mano con cualquier duda.

Índice general

Índice de figuras

Capítulo 1

Introducción

Para comprender la aplicación presentada en este proyecto es necesario conocer los problemas, objetivos, y requerimientos del proyecto multidisciplinar dentro del que se ha concebido. Por ello se presenta inicialmente dicho proyecto, para a continuación presentar la solución desarrollada en este trabajo final de máster.

1.1. Proyecto "Captura, transcripción y gestión multimodal e interactiva de documentos manuscritos"

En la actualidad, aunque las nuevas tecnologías nos rodean, en muchas ocasiones es necesario o más cómodo escribir en papel. Además, existen gran cantidad de textos manuscritos, desde textos antiguos a las notas que podemos escribir día a día, que sería interesante digitalizar. Gracias a los avances del reconocimiento de escritura dicha digitalización se puede realizar de una forma casi automática.

El objetivo de este proyecto consiste en facilitar esta labor, de forma que el usuario pueda escanear rápidamente un documento, editarlo de forma digital, y finalmente volver a imprimirlo si fuese necesario.

Para ello es necesario la unión e interacción de las múltiples técnicas que se presentan en el proyecto. Por una parte es necesario un sistema que permita capturar rápidamente documentos, seguido por el sistema que se encarga de procesarlos y digitalizarlos, y por último también es necesario una interfaz para editar y organizar los documentos. Esta interfaz se materializa principalmente en dos elementos: una mesa de trabajo, una tableta. Una impresora permitirá devolver el documento editado a su medio original.

El objetivo principal del proyecto por lo tanto es dar algunos pasos en dirección al desarrollo e implementación de un sistema interactivo y multimodal para capturar, transcribir y trabajar con documentos en papel, tanto manuscritos como impresos, en un escritorio digital.

1.1.1. Presentación del problema tratado en este TFM

El diseño de la representación de la información y de la interacción de cualquier sistema interactivo es fundamental, pues de él depende que de cara al usuario final el sistema resultante sea útil y facilite la tarea, o por el contrario sea un estorbo que dificulte su trabajo.

Como se ha explicado anteriormente, se requiere un sistema que permita al usuario interaccionar de forma natural, con el fin de reducir al mínimo la brecha entre el mundo analógico y digital, y que permita realizar las tareas de forma eficiente. La interacción mediante gestos multitáctiles ha demostrado ser de gran utilidad, ya que resulta natural y permite a los usuarios realizar tareas de cierta complejidad de forma sencilla y eficiente como se explica, por ejemplo, en los artículos [1] [2] [3].

Se requiere un sistema mediante el cual representar la información digital, y que permita a su vez la coexistencia de documentos en papel con el mundo digital. Este sistema se puede implementar de diferentes formas: con una pantalla convencional, un escritorio físico sobre el que se proyecte la información, una pantalla multitáctil, etc.

Para nuestra tarea es importante también implementar un sistema que permita digitalizar rápidamente los documentos, y volver a convertirlos a su formato físico.

1.2. Objetivos del proyecto

El objetivo principal de esta tesina consiste en el análisis, desarrollo e implementación de dicho Escritorio Digital.

Es necesario analizar las distintas opciones que existen a la hora de implementar este escritorio teniendo en cuenta que debe cumplir los siguientes objetivos.

La interacción se realizará por medio de gestos multitáctiles que serán fáciles e intuitivos. Para que el sistema sea agradable al usuario la interacción se realizará con las manos, evitando periféricos que incomoden.

El sistema debe facilitar la labor del usuario por lo que su usabilidad es muy importante. El sistema debe ser capaz de capturar documentos manuscritos con facilidad y rapidez, estos documentos físicos deben poder coexistir con los digitalizados en el área de trabajo. Los documentos digitalizados deben poder ser trasladados, rotados y escalados por el usuario.

La interfaz debe responder en tiempo real (lo ideal sería que todo el procesado del documento se realizase en tiempo real, pero debido a las necesidades de procesamiento requeridas en la fase de detección es imposible que esta tarea se realice en tiempo real). El sistema debe poder comunicarse y funcionar conjuntamente con las partes realizadas por otros miembros del equipo.

Finalmente el sistema debe ajustarse al presupuesto del proyecto, teniendo un coste económico contenido.

1.3. Descripción de la solución propuesta

La solución propuesta permitirá al usuario manejar de forma eficiente los documentos digitales transcritos, facilitará la tarea de capturarlos y proporcionará además la opción de imprimirlos.

El escritorio permitirá al usuario trasladar, rotar y ampliar los documentos a su gusto para facilitar su visionado. Además proporcionará la posibilidad de archivarlos en carpetas para facilitar la organización de dichos documentos.

No se requerirá ningún tipo de periférico que pueda incomodar al usuario para la interacción con el sistema. El sistema será capaz de detectar el movimiento y gestos que realice el usuario con sus propias manos, y actuar en consecuencia.

La parte relacionada con la corrección interactiva de los textos transcritos se realizará en una tableta digital, ya que la resolución y precisión del escritorio no es suficiente para esta tarea.

En la Figura 1.1 se muestra el primer prototipo de la solución propuesta, que está compuesto por un proyector, una cámara para la detección de las manos y otra cámara para la captura de los documentos. El proyector permite utilizar el sistema en una mesa convencional, en la que pueden coexistir los documentos físicos con los digitalizados. La cámara de captura de las manos permite una interacción natural con el sistema sin utilizar ningún periférico. Una cámara de alta resolución se utiliza para capturar los documentos con buena calidad y rapidez. En la figura falta la tableta para la edición de los documentos, aunque se puede observar junto a más información sobre el sistema final en el Capítulo 5.

1.4. Estructura de esta memoria

En los siguientes apartados se describe, inicialmente un estudio sobre el estado del arte (en el que se analizan antecedentes, y se presentan las bases sobre las que se va a construir la solución). A continuación el análisis del problema, analizando los requisitos, las limitaciones que se presentan, y la forma de afrontar el problema. El diseño de la solución, aportando para ello diagramas UML, y describiendo las distintas partes de las que consta la solución. Por último una muestra de los resultados obtenidos, así como conclusiones y trabajo futuro a realizar.

Finalmente se presentan otros apéndices y bibliografía.

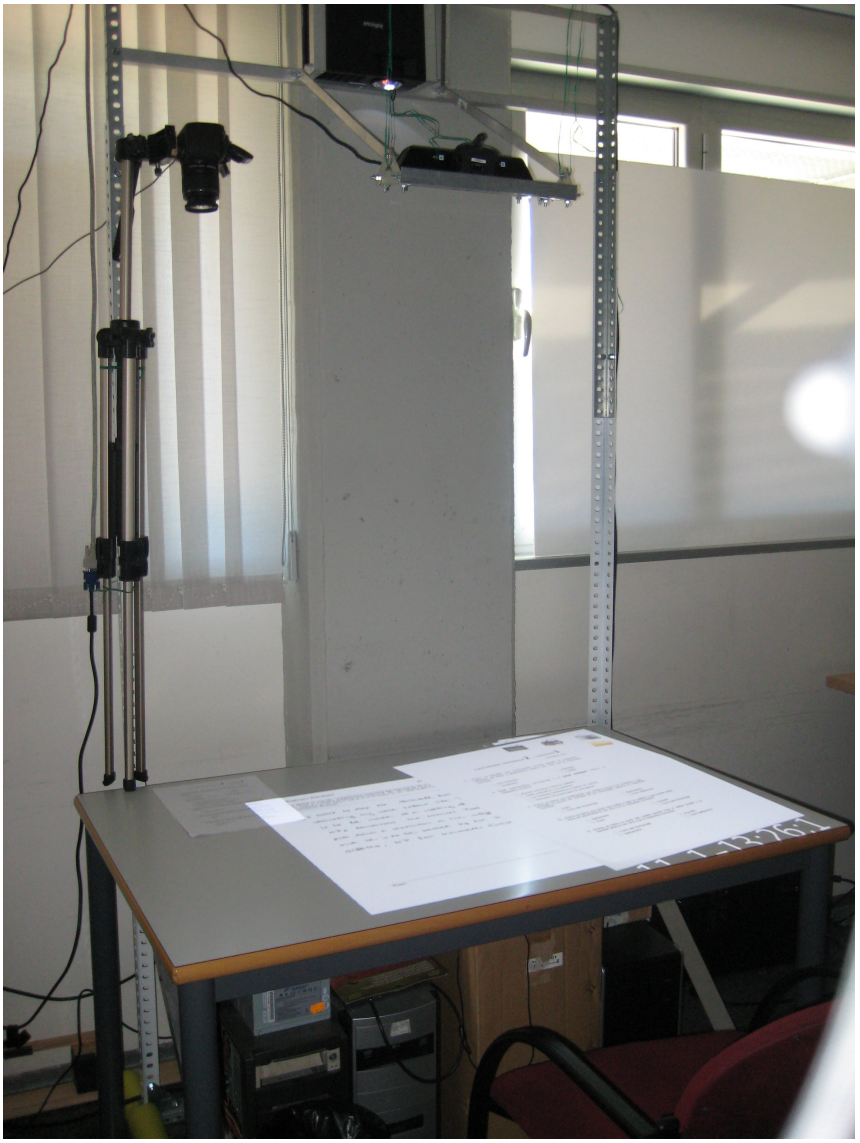


FIGURA 1.1: Se puede observar el prototipo montado en el laboratorio, compuesto por un proyector, una cámara para detección de las manos y una cámara para la captura de documentos. En la mesa se observan dos documentos digitalizados.

Capítulo 2

Estado del arte

En esta sección se realiza un estudio de las diversas soluciones, o aproximaciones al problema existentes. Dado que en el proyecto confluyen varias tecnologías, analizaré cada una por separado en los distintos apartados: escritorios digitales, interacción multitáctil y tecnologías para la detección de múltiples contactos.

2.1. Escritorios digitales

Inicialmente es importante definir el concepto de Escritorio Digital o Escritorio Virtual al que nos referimos en este proyecto, ya que es un término que puede tener una gran cantidad de aplicaciones y por tanto un poco ambiguo. No se trata de mostrar varios escritorios virtuales en una misma pantalla como realizan sistemas operativos como *Mac OS X* y la gran mayoría de distribuciones de *Linux*. Tampoco se trata de la visualización en un "*thin client*" de un escritorio ejecutado por una máquina remota. Por escritorio digital, en este trabajo nos referimos a un espacio de trabajo en el que se muestran los documentos, objetos, etc. virtuales con los que interacciona el usuario, renderizados por un ordenador, simulando el comportamiento de estos elementos en el mundo real.

Existen una gran variedad de propuestas para construir escritorios digitales, con el objetivo de implementar una interacción natural con los elementos del sistema que el usuario puede manejar, tal cual realizaría en un escritorio físico. En la actualidad, gracias fundamentalmente a la interacción multitáctil y técnicas relacionadas con la realidad virtual y aumentada, se ha conseguido acercar bastante el comportamiento de estos escritorios virtuales al de los reales.

A continuación se presentan varios ejemplos de escritorios virtuales, desde los más tradicionales a los más innovadores, con una utilidad más concreta, pero interesantes en cuanto a las posibilidades que ofrecen.

2.1.1. Microsoft® PixelSense™

Se trata de uno de los escritorios virtuales disponibles comercialmente. Antes llamado *Microsoft Surface*, pero rebautizado bajo este nuevo nombre en Junio de 2012 (para pasar a llamar Surface a una nueva *tablet*). Este dispositivo se puso a la venta en su primera versión en el año 2008, y consistía en un cubo que contenía todo el hardware necesario para su funcionamiento. En la cara superior un proyector retroproyectaba la imagen, y un sistema compuesto por un emisor infrarrojo y varias cámaras permitía capturar lo que se posase sobre la pantalla.

FIGURA 2.1: PixelSense 2.0, Samsung SUR40. Una pantalla de alta resolución con sensores integrados permite capturar imágenes de lo que hay sobre la pantalla. Internamente incorpora hardware de PC y un software que permite tratar estas capturas para interaccionar con el sistema.

Actualmente, está disponible la segunda versión de este dispositivo, fabricada y distribuida por Samsung bajo el nombre *Samsung SUR40*, Figura 2.1. Consta de una pantalla de alta resolución, con unos sensores infrarrojos integrados, y una retroiluminación proporcionada por LEDs blancos e infrarrojos, como se muestra en la Figura 2.3. Cuando un objeto se aproxima a la pantalla, este refleja la luz infrarroja, que es capturada por la pantalla. Un software se encarga de analizar la imagen, capturarla e interpretarla para detectar los distintos contactos (ya sean dedos, objetos, u otro tipo de elementos como etiquetas, que el software es capaz de reconocer), como se observa en Figura 2.2.

Se puede utilizar para gran cantidad de aplicaciones, desde visualizar fotografías y vídeos, trabajar de forma colaborativa con varias personas a la vez, e incluso detectar ciertos objetos y responder a ellos.

Este dispositivo tiene un precio aproximado de $8.000. Microsoft proporciona un SDK para facilitar la creación de aplicaciones.

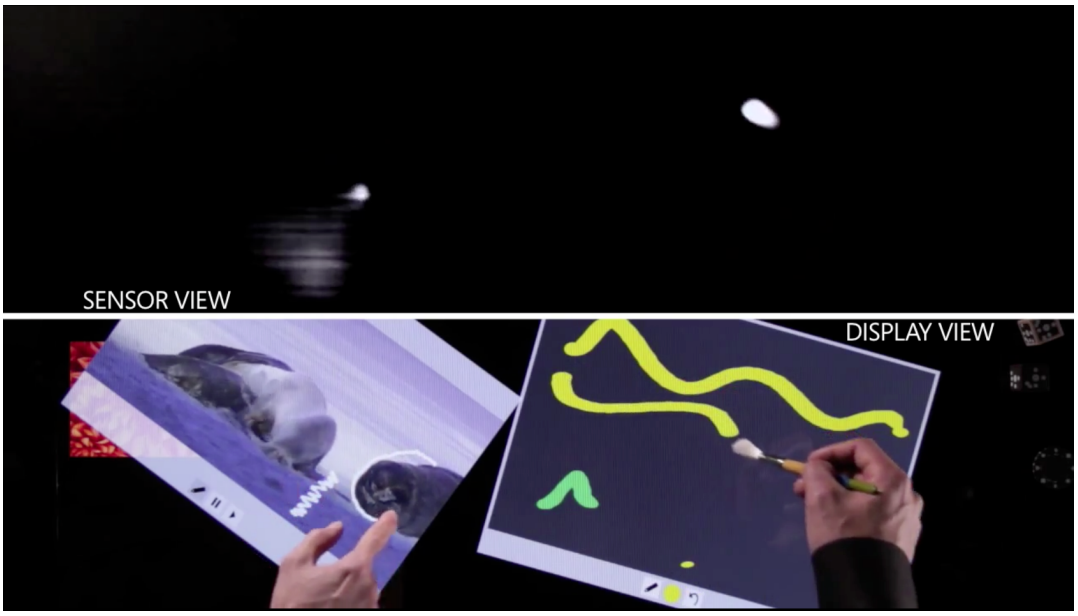


FIGURA 2.2: El sensor de PixelSense 2.0: obtiene una imagen de aquello que se posa sobre la pantalla, ya que los objetos o manos reflejan la luz infrarroja que es captada por los sensores internos (cámaras infrarrojas en la primera versión y el propio panel actúa como una gran cámara en esta última versión).

En relación a esta tesina, es interesante desde el punto de vista que presenta una superficie de trabajo de un tamaño y resolución aceptable (40" y 1920x1080 [5]), compacta, resistente, y un sistema preciso de reconocimiento del contacto con la pantalla. Con un sistema como este se resolvería el problema de la visualización del escritorio y detección de múltiples contactos. El principal problema que presenta es su elevado precio. También habría que considerar si este sistema está pensado para utilizarse como una mesa de trabajo, ya que al contener elementos que desprenden calor (pantalla y componentes de PC) podría ser incómodo, y además se requeriría algún dispositivo extra para poder capturar los documentos manuscritos.

2.1.2. nSquared Presenter 2.0

Se trata de un software comercial desarrollado sobre la plataforma anterior (Samsung SUR 40). Permite trabajar con documentos y distintas fuentes (PC, Mac, tabletas), de forma que se puede crear una presentación con imágenes, documentos, vídeos, etc. y transferirlos de forma sencilla a otros ordenadores, tabletas, o proyectores.

En el vídeo [6] se puede observar su funcionamiento, y la naturalidad con la que se trabaja con documentos en una superficie multitáctil, pudiendo por ejemplo ampliarlos para visualizarlos correctamente y realizar anotaciones sobre ellos.

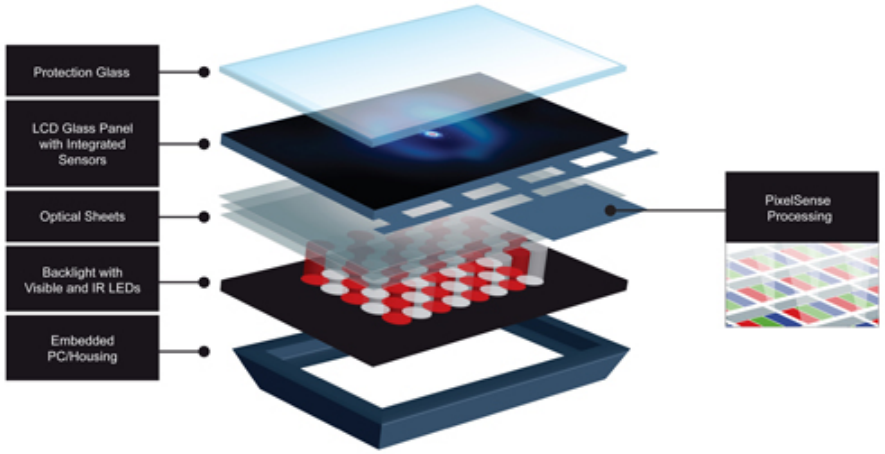


FIGURA 2.3: Componentes internos de PixelSense 2.0: Partes del panel responsable de la visualización y captura. De arriba hacia abajo: cristal protector, panel LCD con sensores ópticos integrados, láminas que se encargan de dirigir la luz producida por el siguiente panel, el de retroiluminación, y finalmente una carcasa que aloja los componentes de PC que se encargan de manejar todo el conjunto [4].

Desde el punto de vista de esta tesina, es interesante la forma en la que se utiliza el sistema con distintos dispositivos, algo similar a lo que se persigue en este proyecto al utilizar junto al escritorio digital una tablet que permita editar con precisión los documentos. Es interesante también la interacción multitáctil, puesto que se muestra como una forma natural para trabajar con documentos. Dado que el sistema sobre el que se ejecuta el software es el presentado anteriormente en la Sección 2.1.1, presenta los mismos inconvenientes.

2.1.3. PlayAnywhere

Este sistema ideado por *Andrew D. Wilson* [7], muestra la posibilidad de conseguir una superficie interactiva sobre una mesa utilizando un proyector compacto y un sistema de captura compuesto por un emisor infrarrojo y una cámara sensible al infrarrojo montado sobre dicho proyector, se puede observar en la Figura 2.4. Además, también se detallan en él posibles formas de reconocer la interacción del usuario con el sistema, ya sea reconociendo las sombras que produce la mano al cortar la luz infrarroja, o reconocer directamente ciertos objetos como pueden ser hojas de papel, ciertos patrones que permiten identificar objetos (al igual que los marcadores utilizados en realidad aumentada) o detectar el flujo óptico del movimiento y utilizarlo para interaccionar con el sistema.

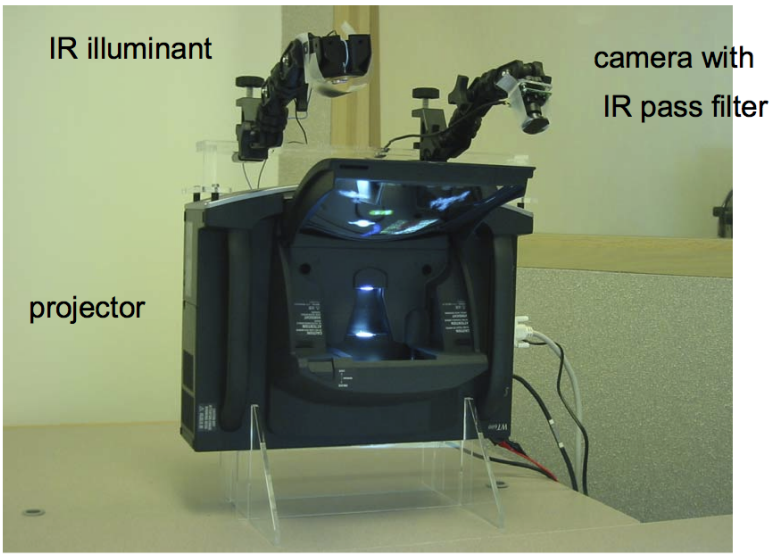


FIGURA 2.4: Componentes del sistema PlayAnywhere. Se observa principalmente un proyector compacto (que utiliza un espejo para proyectar su imagen sobre una superficie cercana) y situados sobre el proyector los elementos encargados de la captura, un emisor de luz infrarroja y una cámara a la que se le ha puesto un filtro que sólo deja pasar luz infrarroja [7].

Este artículo es muy interesante desde el punto de vista de este trabajo final de máster, ya que presenta un sistema parecido al que podemos conseguir en el laboratorio. En él destacan el uso de un proyector compacto que permite proyectar sobre la propia mesa en la que se está apoyando, lo que permite que el sistema sea fácilmente transportable. La principal ventaja de utilizar un proyector situado sobre la mesa es que su parte inferior queda vacía, de forma que el usuario puede sentarse cómodamente. El uso de una cámara y un proyector infrarrojo permite realizar el reconocimiento sin importar la luz del propio proyector que incide sobre ellas. Si se sustituye el proyector por uno convencional, el sistema pasaría a tener un coste económico reducido, aunque se perdería la facilidad de transporte, lo que no es un requisito necesario para este proyecto.

En el artículo se presenta también la posibilidad de detectar páginas, que puede ser también de interés para este proyecto, ya que se va a trabajar con documentos.

Dado que se captura la posición de las manos sobre la mesa con una cámara situada en el proyector, existen casos que serían reconocidos incorrectamente, desde objetos que imiten la forma de las manos, a casos en los que una mano haga sombra sobre la otra y sea por lo tanto indetectable.

También se seguiría necesitando una cámara adicional para poder capturar a alta resolución los documentos, ya que la utilizada para capturar las manos, dada la limitada

resolución junto con la distorsión por su ubicación, es incapaz de capturar imágenes de calidad.

2.1.4. EnhancedDesk

El sistema descrito en [8] ya en el año 1998, muestra las posibilidades de utilizar un sistema de proyección junto a visión por computador para aumentar, de forma digital, la información disponible en documentos físicos. En este caso utilizan técnicas como las empleadas en realidad aumentada para, mediante la utilización de marcadores, ampliar la información disponible en ciertos documentos. Además, el sistema es capaz de detectar la interacción del usuario y responder mostrando más información sobre los puntos que toca, e incluso sobre un mapa es capaz de detectar varios dedos y mostrar la ruta más corta entre ambos puntos. Se puede observar un diagrama de su funcionamiento en la Figura 2.5.

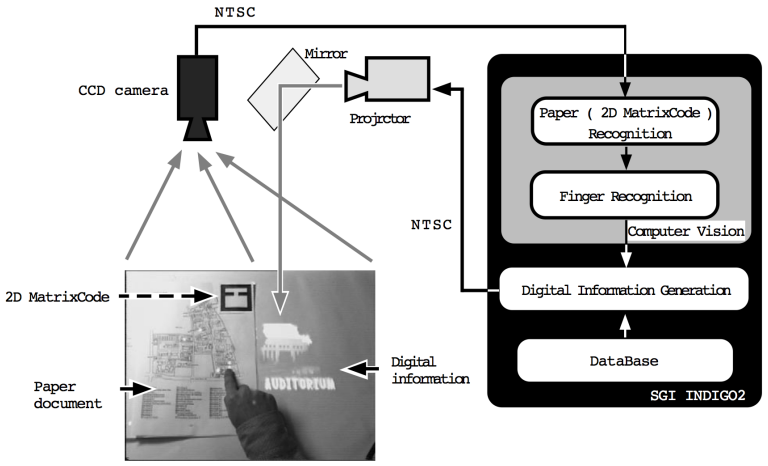


FIGURA 2.5: Arquitectura de EnhancedDesk. Una cámara y un proyector situados sobre la mesa se encargan de capturar y mostrar la información. Los documentos están etiquetados con un código definido en una matriz 2D y el reconocimiento de dedos se realiza reconociendo el color de la piel [8].

Este artículo presenta uno de los primeros acercamientos a lo que se pretende conseguir con este proyecto. Tiene muchas limitaciones: principalmente, no reconoce los documentos (sino que simplemente detecta y reconoce un código que los identifica), se utiliza el color de las manos para detectarlas (por lo que presentará problemas cuando cambien las condiciones de iluminación de estas, como al pasar bajo la imagen proyectada). Es interesante desde el punto de vista en que permite la coexistencia de un documento en formato físico junto con la información digital que presenta la aplicación.

2.1.5. Keepin' It Real: Pushing the Desktop Metaphor with Physics, Piles and the Pen

En este trabajo de *Agarawala y otros* [9], se introduce la posibilidad de realizar un escritorio virtual que actúe como un escritorio físico. En él los documentos aparecen como iconos que se pueden apilar o desapilar, mover, y rotar por el escritorio (como se puede observar en la Figura 2.6). También, una vez abiertas, las aplicaciones se pueden manejar como si fuesen documentos u objetos en el mundo real. El sistema está pensado para ser utilizado en un *tablet* con un puntero, por ello la interacción se ha optimizado para este tipo de entorno evitando elementos pequeños y acciones difíciles como doble click, etc.

Se ha centrado el proyecto en que el comportamiento y apariencia del sistema fuese parecido al de los objetos reales, proporcionándoles un comportamiento físico, una apariencia parecida al papel a los iconos, aunque sin que esto entorpezca su uso, deshabilitando en ciertos casos por ejemplo el comportamiento físico de los objetos.

FIGURA 2.6: Escritorio con física y pilas. En la imagen se pueden observar ventanas de distintos programas, así como imágenes y documentos. Estos se comportan como si fuesen documentos en el mundo real y se pueden mover y organizar arrastrándolos o manipulándolos con ciertos gestos predefinidos ideados para realizarse con un puntero [9].

Este proyecto presenta cómo el uso de simulación de comportamientos físicos naturales, a la hora de tratar documentos, apilarlos, organizarlos, etc. es interesante desde el punto de vista de un escritorio virtual. Dado que está pensado para utilizarse con un único puntero y en una superficie pequeña como es una tablet, los gestos no son relevantes para el proyecto que estoy presentando, pero sí que lo es la idea de poder tratar los documentos como si fuesen documentos en el mundo real.

2.1.6. Toucheo

En este sistema descrito en [3], se plantea la posibilidad de utilizar paneles multitáctiles y pantallas estereoscópicas, de forma que el usuario puede visualizar un objeto en 3 dimensiones y manipularlo utilizando gestos multitáctiles. El sistema es interesante ya que permite la coexistencia de elementos físicos (las manos) con elementos digitales, utilizando para ello un espejo semitransparente, sobre el que se proyecta la imagen mientras que permite ver las manos que están bajo él.

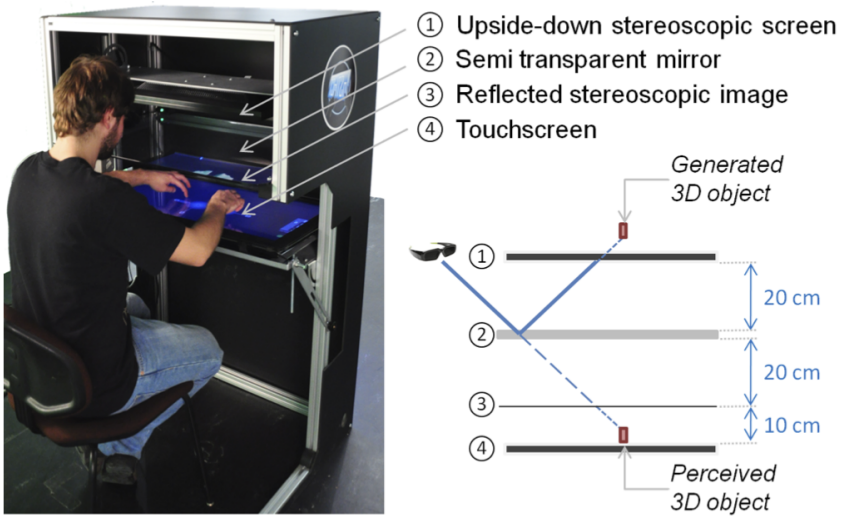


FIGURA 2.7: Toucheo, componentes físicos del sistema. De arriba hacia abajo, un monitor estereoscópico situado apuntando hacia abajo, un espejo semitransparente que refleja la imagen de la pantalla, bajo el espejo el volumen sobre el que el usuario ve los objetos, y finalmente el panel multitáctil [3].

Este artículo presenta un tipo de escritorio virtual orientado a la manipulación de objetos en 3D. Es interesante desde el punto de vista de esta tesina, ya que permite la coexistencia de elementos físicos y digitales y gracias al espejo semitransparente, las oclusiones de los objetos físicos sobre los digitales no impiden la visualización de ambos. Uno de los problemas que presenta es que el panel capacitivo que actúa de sensor multitáctil tiene un determinado tamaño, un coste elevado, además de no ser el material más idóneo para trabajar sobre él con documentos en papel. También el montaje físico del sistema, aunque es mucho más compacto que el utilizado en otros artículos en los que se emplean proyectores, necesita estar encerrado en el armazón que se observa en la Figura 2.7 ya que la visibilidad del sistema no es muy buena debido a que se pierde gran cantidad de iluminación debido a que el espejo es semitransparente y no refleja toda la luz del monitor y se utiliza 3D activo, con gafas, que también reducen la cantidad de luz que llega al ojo. Este tipo de armazón dificultaría su uso para manipular documentos como se pretende hacer en el proyecto.

2.1.7. BendDesk

En el trabajo de *Weiss y otros* [10], se presenta un tipo de escritorio caracterizado por disponer de una pantalla *curvada* 90º en la que el usuario puede utilizar parte de la pantalla en vertical y parte en horizontal.

Los componentes del sistema se pueden observar en la Figura 2.8, donde se observa vista de perfil la pantalla curva. En el artículo se explica con detalle los diversos problemas y soluciones que se han aplicado para conseguir que el sistema funcione adecuadamente y sea capaz de proyectar imágenes y detectar pulsaciones sobre la curva con precisión. Los autores observan, que la parte curva de la pantalla actúa como una barrera, y los usuarios evitan utilizarla (y realizan movimientos más lentos cuando tienen que arrastrar un objeto sobre esa parte de la pantalla). Diseños de este tipo deberían evitar hacer al usuario cruzar continuamente ese área (podría utilizarse como *Dock* o *Taskbar* para almacenar temporalmente objetos entre ambas partes).

Desde el punto de vista de esta tesina, es interesante ya que presenta un tipo distinto de escritorio virtual. La pantalla curva podría ser interesante en un escritorio virtual para tratar documentos, ya que permite tener una superficie horizontal sobre la que trabajar con documentos físicos y virtuales, y una vertical que puede ser útil para visualizarlos. El problema que presentan los paneles basados en la tecnología *FTIR* (como el presentado en la UPV [11]) es que el acrílico podría presentar problemas de rigidez e impedir trabajar con documentos en papel sobre su superficie.

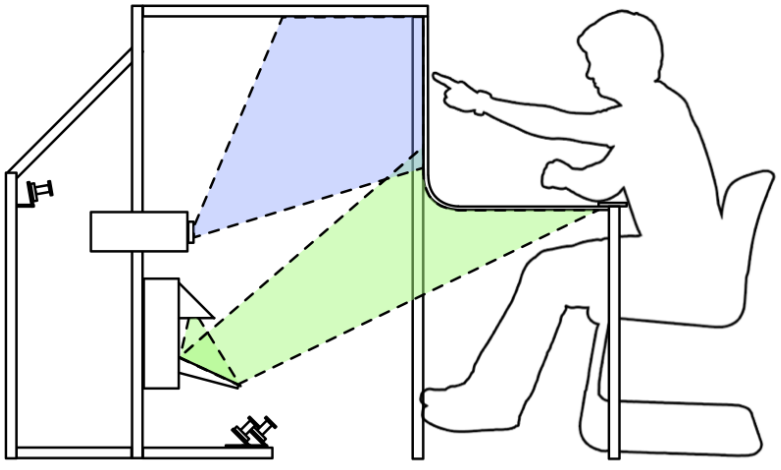


FIGURA 2.8: BendDesk, componentes físicos del sistema. Se puede observar la estructura que soporta el sistema. Contiene los dos proyectores y tres cámaras para captar el contacto con los dedos, el panel utiliza tecnología *FTIR*[12] con una lámina de material acrílico con LEDs infrarrojos situados alrededor [10].

2.1.8. Otros

Existen muchos más artículos que muestran escritorios, cada uno de ellos con ciertas características que lo diferencian de los anteriores. Por no extender mucho más esta sección, aquí se describen brevemente algunos de ellos.

En [13] se describe un sistema llamado *WeSearch*, que propone un sistema colaborativo de búsqueda web en el que varios usuarios interaccionan simultáneamente con una gran mesa (que utiliza un sistema de captura de dedos mediante luz infrarroja y cámaras). En este sistema los usuarios manipulan lo que llaman *clips*, que son contenido web renderizado en pequeñas ventanas. Se puede observar la apariencia del escritorio en la Figura 2.9.

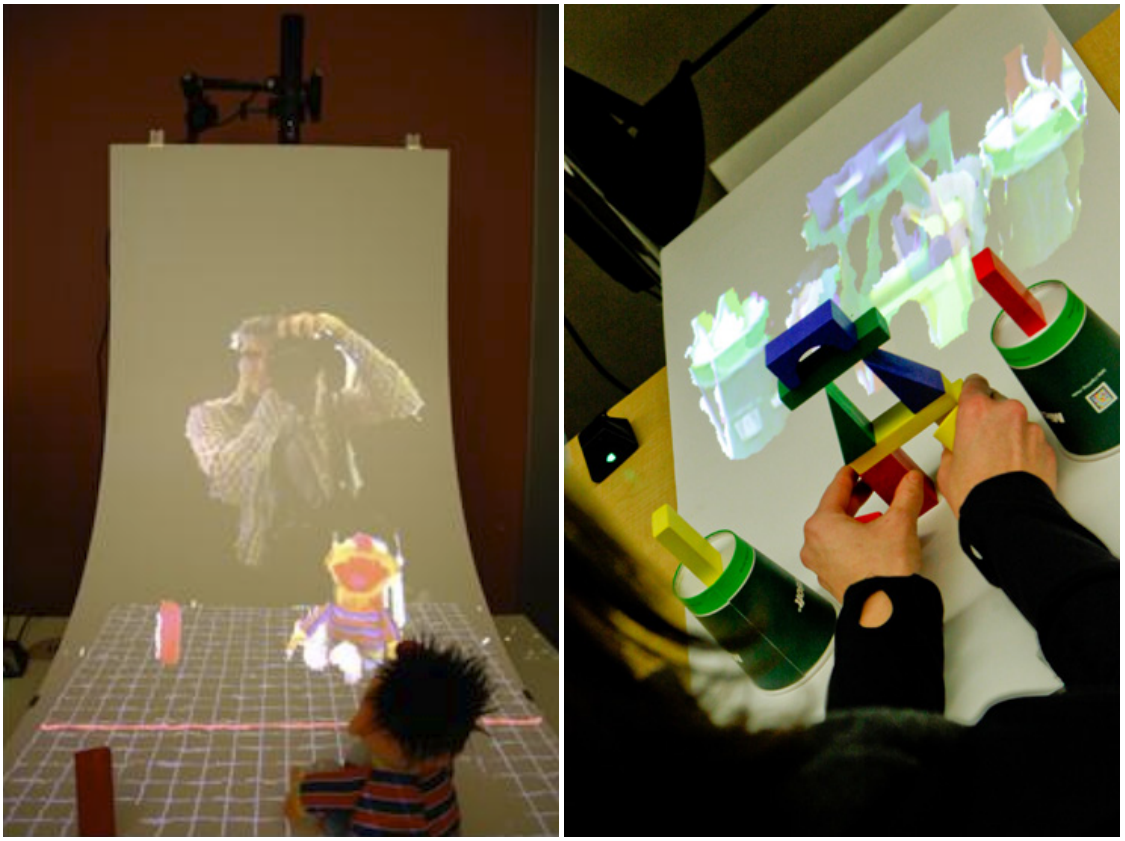
FIGURA 2.9: Escritorio WeSearch, en él varias personas pueden colaborar manejando una gran mesa.

Este sistema es interesante ya que se utiliza una gran superficie (1,2m x 1,8m) y dos proyectores para mostrar la información. Pero el reconocimiento de contactos se realiza bajo la mesa, lo cual presenta el problema de que no se puede utilizar como una mesa de trabajo, puesto que no se pueden introducir las piernas sin entorpecer su funcionamiento.

También existen sistemas más antiguos que utilizan "trackers" para registrar el movimiento del usuario, y unas gafas estereoscópicas capaces de mostrar no sólo imagen digital sino también hacerse transparentes para mostrar el mundo físico, de forma que permiten al usuario trabajar con objetos virtuales como si fuesen reales. Se puede ver un ejemplo ejemplo en el artículo [14].

En [15] y [16] se describen dos proyectos relacionados, el primero *MirageTable*, y el segundo *StereoBlocks* (también llamado *MirageBlocks* [17]), se basan en ofrecer al usuario

una mesa de trabajo sobre la que al poner un objeto este se puede capturar y manipular de forma digital. La visualización es 3D gracias a un proyector y unas gafas. El registrado en 3D que se consigue con una cámara de profundidad y otra de color como las que tiene Kinect. Se pueden ver unas capturas en Figura 2.10.



(A) MirageTable (B) MirageBlocks

FIGURA 2.10: En ambos el sistema puede capturar el objeto físico en 3D, que posteriormente se puede manipular de forma digital. En MirageTable (A) se utiliza una pantalla curva, el sistema puede actuar como un espejo como se observa en la imagen, o capturar objetos e interaccionar con ellos. En MirageBlocks (B) se utiliza una superficie plana sobre la que se pueden capturar, por ejemplo, *bloques* para construir escenas virtuales.

En *ARWin* [18], se utiliza realidad aumentada para unir el mundo físico con el virtual en un escritorio que es capaz de mostrar ventanas de aplicaciones (ejecutadas en un servidor VNC y mostradas como una textura sobre un cuadrado), el usuario se pone unas gafas y además es necesario situar marcadores en el escritorio físico para que el sistema pueda reconocer las diferentes zonas. Se puede observar una imagen del sistema en la Figura 2.11.

FIGURA 2.11: Escritorio ARWin, se utilizan marcadores y técnicas de realidad aumentada para situar sobre un escritorio físico elementos digitales, como pueden ser objetos 3D y ventanas de ciertas aplicaciones.

2.2. Interacción multitáctil

En este apartado se presentan algunos de los artículos relevantes respecto a la interacción multitáctil. Una vez vistas las posibilidades que permite un escritorio virtual, hay que analizar cómo interaccionar con él, con los objetos y documentos que nos presenta, de forma sencilla y eficiente. Se analizan artículos que principalmente tratan sobre los gestos que son más naturales de cara al usuario, cómo detectarlos y hacer que nuestro sistema actúe en consecuencia.

2.2.1. User-defined gestures for surface computing

En [1], se presentan multitud de observaciones al respecto de cómo usuarios noveles actúan frente a un entorno multitáctil cuando se les pide realizar ciertas acciones. En su experimento se seleccionaron 20 participantes que nunca habían utilizado un dispositivo multitáctil ni eran expertos en informática.

Utilizando un prototipo de Microsoft Surface (ahora PixelSense) y una aplicación diseñada para el experimento, se les pedía realizar 27 acciones (desde mover, abrir, o pasar hojas en documentos a otros más complejos como deshacer, minimizar, cortar, etc.). Para ello la aplicación primero exponía al usuario la acción a realizar (por ejemplo, "Gire la cámara para revelar el objeto que hay fuera de la pantalla"). Tras esto se veía una animación del efecto del gesto y volvía a su posición inicial esperando la interacción del usuario. Se pedía al usuario que realizase el gesto con una mano y luego con dos manos.

Se grababan tanto los puntos de contacto, como las manos del usuario. Finalmente el usuario tenía que valorar si prefería realizarlo con una o dos manos y valorar en dos escalas si el gesto era adecuado y sencillo.

Utilizando todos estos datos, se obtuvieron 1080 gestos (20 personas · 27 gestos · 2 modos), y en base a los resultados y su análisis se obtuvo un conjunto de gestos definidos por los usuarios, que es el que se puede observar en la Figura 2.12

A partir del análisis de los datos obtenidos se obtuvieron varias conclusiones: generalmente los usuarios prefieren gestos de una mano (aunque existen ciertas acciones sobre las que se prefieren los de dos manos), creen que el gesto es adecuado si se puede pensar más rápidamente y consideran más fácil aquellos que se articulan en menor tiempo.

Otras conclusiones que se desprenden del estudio, es que los usuarios relacionan conceptos similares a gestos similares (zoom y agrandar, mover objetos y girar la cámara), se pueden utilizar los mismos gestos (simplificando de esta forma la facilidad de cara al usuario de memorizarlos) simplemente distinguiendo si se realizan sobre el objeto o el fondo. El

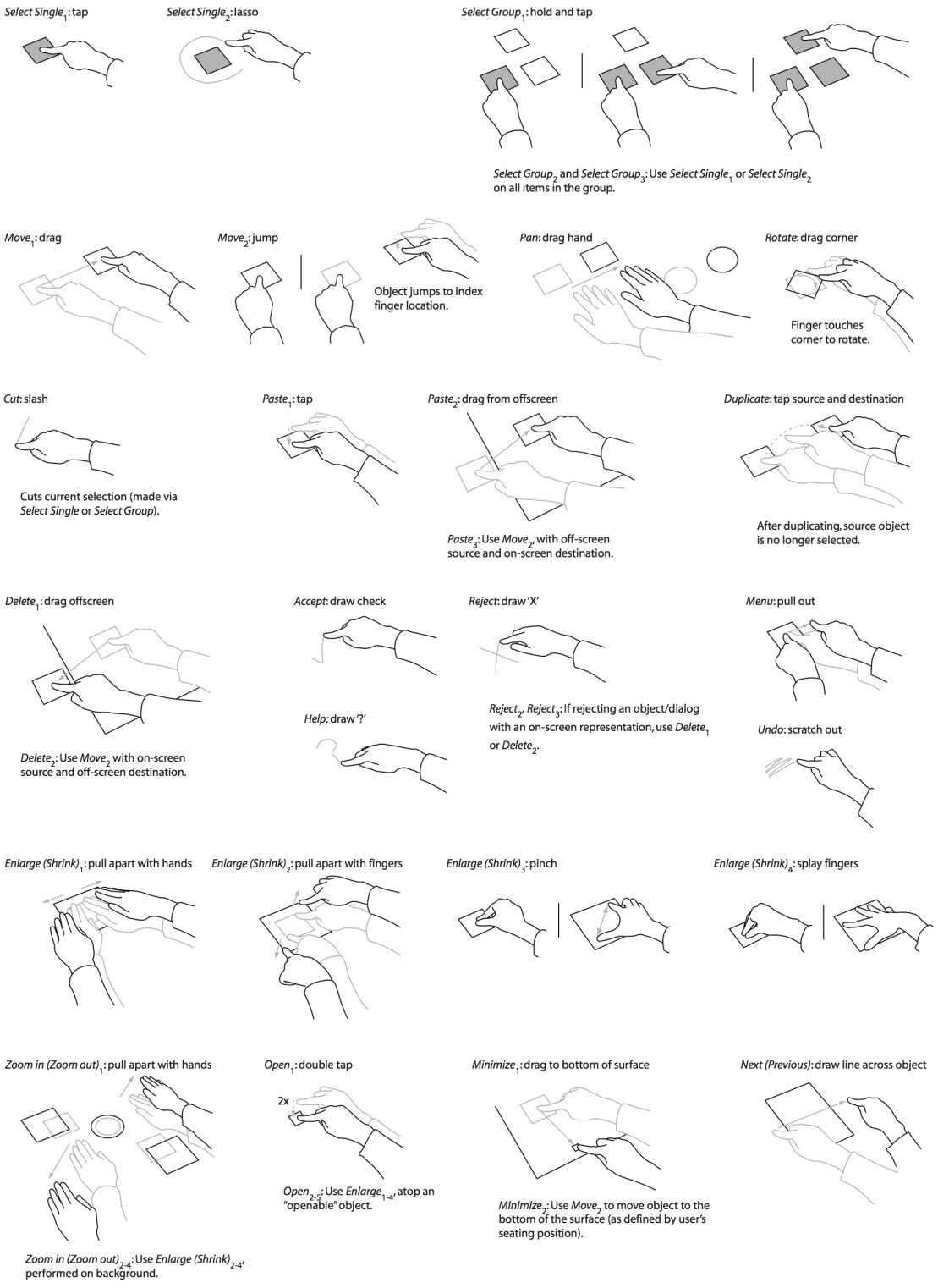


FIGURA 2.12: El conjunto de gestos multitáctiles. No se muestran todos los obtenidos ya que muchos se realizan igual pero en dirección contraria, o sobre el fondo en vez de objetos. Los que muestran un dedo pueden realizarse indistintamente con uno o varios dedos [1].

número de dedos empleados es variable. Generalmente los usuarios utilizan más dedos para objetos más grandes (se deduce de ello que el número de dedos no debería ser una variable que distinga el tipo de gesto, ya que en el mundo real las personas no prestan atención a cuántos dedos utilizan).

Además, pese al esfuerzo de camuflar el sistema operativo interno, los usuarios tratan el sistema como si fuese un entorno de ventanas, realizando ciertas acciones y gestos como si el objeto fuese una ventana en Windows. Por ejemplo se observa que algunos usuarios realizaban *clicks* con el dedo índice y con el corazón para emular el *click* izquierdo y derecho del ratón, también muchos usuarios realizaban un primer *click* para seleccionar el objeto y posteriormente realizaban el gesto en vez de aprovechar la posibilidad de seleccionar e interaccionar con el objeto en el mismo gesto. La acción de *cerrar* la realizaban presionando la esquina superior derecha de la ventana, y para cambiar de tarea usaban la parte inferior de la pantalla.

2.2.2. Eden: a professional multitouch tool for constructing virtual organic environments

En [2], se presenta un sistema de interacción multitáctil pensado para ayudar a diseñadores gráficos a construir escenas 3D. Se presentan para ello un conjunto de gestos que permiten realizar las operaciones básicas de rotación, escalado y traslación, y se han diseñado teniendo en mente su simplicidad.

Utilizando este sistema se pidió a un diseñador profesional experimentado que lo utilizase reproduciendo una escena previamente diseñada, y los resultados fueron bastante favorables. El sistema presenta ventajas a la hora de manipular objetos, se compara con el software de diseño Maya, y frente a este, el sistema permite manipular de forma simple y rápida objetos sin necesidad de seleccionarlos y manipularlos con un pequeño manipulador como ocurre en Maya. Aunque el gesto de traslación presenta problemas ya que los objetos pequeños se ocluyen con el dedo. Simplifica también el proceso de manipular la cámara y añadir objetos. Se estima que el sistema ayuda a realizar el trabajo un 20 % más rápido.

Los gestos que se proponen para manipular los elementos se muestran en la Figura 2.13. El sistema diferencia entre gestos relacionados con uno o dos dedos.

Del análisis del experimento se obtienen varias conclusiones, muchas de ellas en común con el artículo anterior (como la reutilización de gestos, interpretar los gestos según dónde se realicen, etc) y otras nuevas, como el tener en cuenta la oclusión que presentan

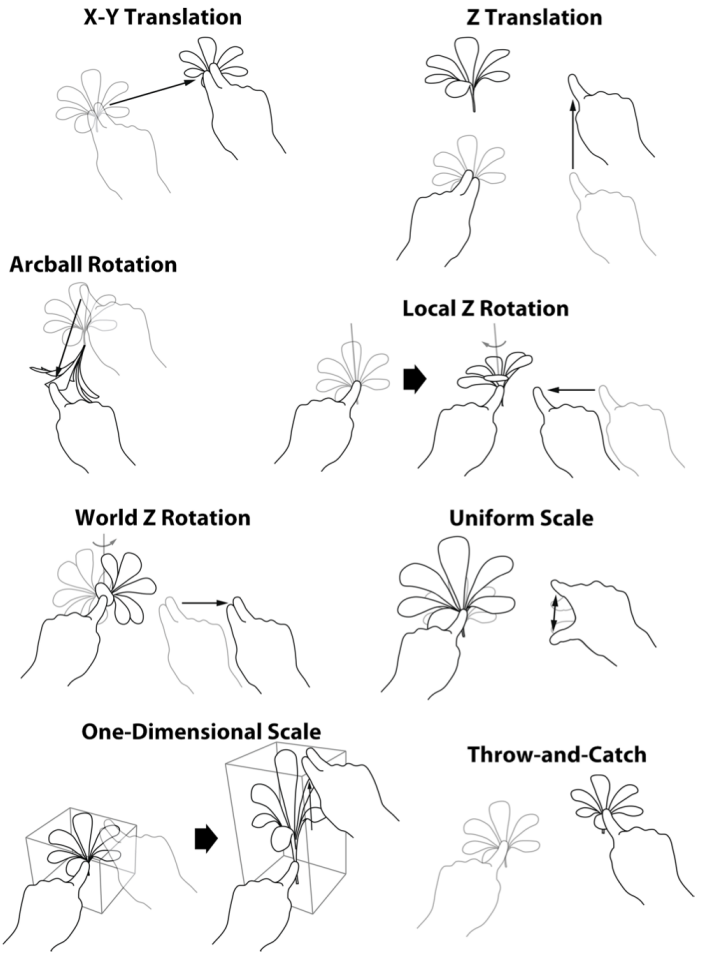


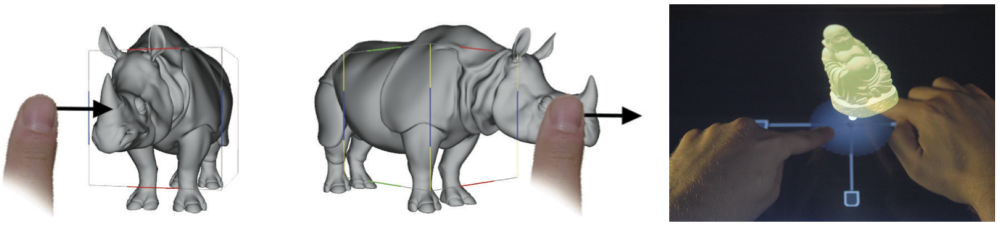
FIGURA 2.13: Gestos de Eden. Se utiliza el toque con dos dedos juntos para diferenciar entre ciertos gestos [2].

los dedos en objetos, además de intentar mantener un sistema lo más sencillo posible, realizando una acción por gesto, ya que esto facilita al usuario su comprensión.

2.2.3. Otras formas de manipulación, utilizando "Widgets"

En artículos como [19] y [3], se presenta un sistema de interacción en el que al tocar un objeto aparece junto a él lo que denominan "Widget", que es básicamente un elemento (los bordes de una caja que rodea al objeto en el primero, Figura 2.14a, y unos ejes en el segundo, Figura 2.14b) que permite la manipulación del objeto de una forma clara, ya que al tocar las distintas partes del "widget" se intuye cómo va a reaccionar el objeto.

Estos *widgets* pueden manipularse de forma multitáctil (por ejemplo en Toucheo, si se separan dos dedos tocando la base del objeto este se amplía), o de una forma más tradicional tocando los distintos componentes del *widget* con un sólo dedo.



(A) tBox Widget (B) Toucheo Widget

FIGURA 2.14: Widgets de manipulación 3D. (A) muestra los widgets utilizados en el artículo *tBox [19]* que consisten en una caja que cubre el modelo. (B) muestra el widget utilizado en el artículo *Toucheo [3]* que consiste en unos ejes y una base situados bajo el modelo.

2.2.4. Otros

En el trabajo de *Hancock y otros* [20], se presenta un sistema de manipulación que permite al usuario manipular objetos con 3 dedos. Además se presenta lo que llaman "Sticky Tools", que básicamente supone asignar a objetos distintas acciones que permiten manipular otros objetos 3D de la escena. Por ejemplo se puede utilizar un pañuelo para colorear objetos (de forma que al arrastrarlo sobre un determinado objeto, toma el color del pañuelo), un cajón que permite escalar con un dial cualquier objeto que se deposite sobre él, o también un cajón con texturas que permite pintar con ellas diversos objetos.

En el artículo de *Reisman y otros* [21], se añaden más formas de interacción, ampliando con tres y cuatro dedos las posibilidades de interacción multitáctil con elementos 3D, para por ejemplo rotar el objeto en un determinado plano, o manipular superficies planas en cualquier dirección con un mismo gesto.

2.3. Detección multitáctil

En este apartado se estudian las posibilidades para capturar el movimiento de las manos, y principalmente de los dedos, para conseguir detectar los gestos mencionados en el apartado anterior. Se excluyen de este análisis dispositivos como punteros, o "trackers" que supongan utilizar algún elemento extra ya que se persigue para la tesina que la interfaz sea lo más natural y amigable posible.

2.3.1. Utilizando paneles sensibles al contacto

Es la forma común con la que interaccionamos actualmente con la gran mayoría de dispositivos táctiles. Su funcionamiento se basa en el uso de un panel que es capaz de detectar la posición de los puntos de contacto con la piel del usuario.

El funcionamiento general de este tipo de paneles se describe en el artículo de *Dietz* [22], que ya en el año 2001 mostraba cómo el uso de una matriz de antenas capaz de medir la capacitancia se podía utilizar para detectar los toques sobre una superficie. Además, en este artículo se presenta un sistema mediante el cual, poniendo un receptor en la silla sobre la que se sentaba la persona, era capaz de distinguir quién estaba tocando la superficie y así poder diferenciar entre dos usuarios que podrían tocar varios puntos de la superficie simultáneamente.

Este tipo de paneles son los utilizados hoy en día en la gran mayoría de dispositivos multitáctiles modernos, como teléfonos móviles y tabletas. En *Toucheo* [3], se utiliza un panel de este tipo.

2.3.2. Utilizando cámaras y visión por computador

Dentro de este tipo de sensores existen diversas técnicas y configuraciones. Podemos distinguir entre el tipo de sensor, según sea una cámara RGB convencional, una cámara de infrarrojos, cámaras estéreo, o una cámara de profundidad. También según la forma de capturar el contacto, pues depende de si la cámara está situada bajo la superficie o sobre ella. A continuación se analizarán alguno de estos sistemas.

La ventaja que presentan estos dispositivos reside en que se pueden utilizar sobre superficies de prácticamente cualquier tamaño, y su coste es bastante reducido.

Utilizándo cámaras RGB convencionales

Es la solución utilizada en el artículo [8], *EnhancedDesk*, utilizando una cámara y un software de visión por computador capaz de detectar los dedos, basándose en el color de la piel.

Este tipo de sistema puede utilizarse en entornos con iluminación controlada, pero presenta ciertos problemas ya que la propia iluminación del proyector puede hacer difícil detectar con precisión el dedo en ciertas circunstancias, o ciertos elementos de la escena podrían confundirse con dedos si tienen un color similar. Además, es complicado detectar el punto en el que el dedo hace contacto con la superficie.

Utilizando cámaras infrarrojas

Este sistema utiliza generalmente una lámpara de infrarrojos, que permite proyectar un claro haz de infrarrojos sobre la superficie, y una cámara capaz de captar este tipo de luz (junto con un filtro para evitar que capte el resto del espectro). El sistema puede montarse sobre la superficie o bajo ella.

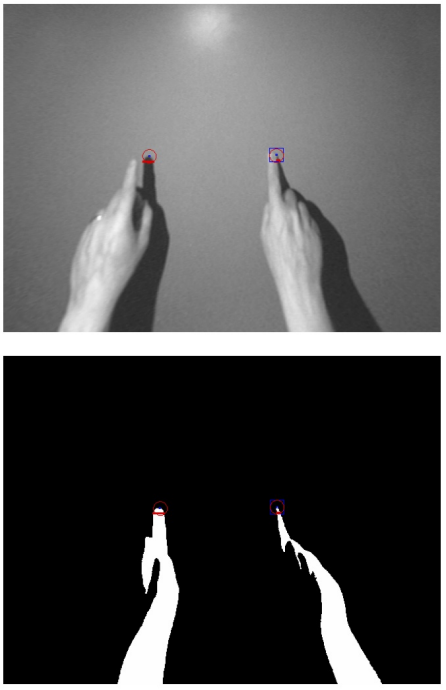


FIGURA 2.15: Detección con cámara infrarroja: se observa la imagen del artículo *PlayAnywhere* [7], donde se detectan los dedos según su sombra, la forma puntiaguda del dedo de la mano derecha indica que está tocando la mesa.

La configuración sobre la superficie se puede observar en el sistema *PlayAnywhere* [7], donde se implementó un reconocedor que detecta la sombra que produce la mano sobre la superficie. Esta sombra tiene una forma redondeada cuando el dedo está flotando y una forma puntiaguda cuando está en contacto con la superficie. Se puede observar en la Figura 2.15. De esta forma, el sistema es robusto frente a la luz que proviene del proyector u otras fuentes, no como en el caso anterior. Sin embargo, presenta problemas por ejemplo para detectar ciertos gestos donde los dedos se cruzan (este tipo de problema sucede en todos los sistemas que utilizan cámaras sobre la superficie), y problemas con elementos que producen una forma similar a la de un dedo.

Bajo la superficie, como el utilizado en la primera versión de *Microsoft PixelSense* (antes *Surface*) [23], también en la actual versión de este dispositivo de una forma más avanzada (la cámara está integrada en el propio panel, Figura 2.3), o el utilizado en la mesa

multitáctil construida en la UPV [11], entre otros muchos proyectos que utilizan técnicas similares. Aunque la tecnología utilizada en cada uno es diferente, todos se basan fundamentalmente en el reflejo o distorsión que se produce en la luz infrarroja al tocar el panel. Al estar las cámaras bajo dicho panel, no presenta problemas de oclusión como el anterior (pero tampoco se puede utilizar el hueco que hay bajo la propia pantalla, así que esto dificultaría el utilizarlo en una mesa de trabajo, como en el proyecto de esta tesina, ya que complicaría meter las piernas bajo la mesa).

Utilizando cámaras estéreo

En el artículo escrito por *Agarwal et al.* [24], se muestra un sistema que es capaz de reconocer con precisión el contacto de los dedos con una pantalla utilizando una cámara estéreo. Para eliminar el fondo de la pantalla, se utiliza un filtro polarizado, ya que la pantalla LCD emite luz polarizada, de esta forma se elimina totalmente la luz proveniente de la pantalla (esta aparece como apagada ante la cámara), y se puede detectar los dedos u otros objetos con facilidad utilizando técnicas de "Machine Learning".

Utilizando cámaras de profundidad

Gracias a dispositivos como *Kinect* [25] que proporcionan este típo de cámaras a un precio asequible, este tipo de tecnología ha ganado popularidad. En el artículo de *Andrew D. Wilson* [26] se presenta una forma sencilla de detección de contacto utilizando este sistema. Una ventaja de la tecnología empleada es que puede detectar contactos sobre cualquier tipo de superficie, ya sea plana o una superficie irregular. El funcionamiento se puede observar en proyectos como *MirageTable* [15] donde se utiliza entre otras cosas para reconocer la interacción con el usuario. Ha de utilizarse sobre la superficie, por lo que presenta el mismo problema de oclusión que otros dispositivos que se sitúan sobre la superficie.

2.3.3. Utilizando Kinect

A causa del interés que ha despertado en los últimos años, hemos creído conveniente analizar más detalladamente el funcionamiento, ventajas y limitaciones que presenta este dispositivo.

Principalmente cuenta, como se observa en la Figura 2.16, con dos sensores de imagen: una cámara RGB, y una cámara de profundidad (también cuenta con más sensores, como micrófonos y sensores de inclinación, pero son irrelevantes para el propósito del proyecto). Estas cámaras tienen las siguientes características [27]:

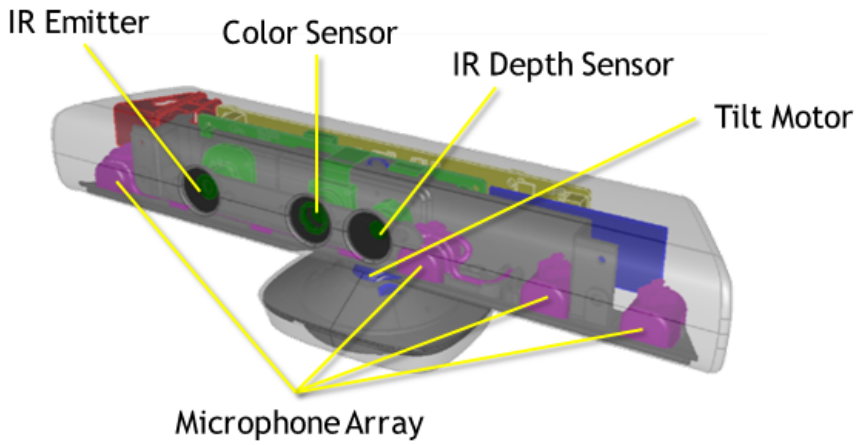


FIGURA 2.16: Componentes de Kinect

- Cámara RGB: resolución desde 640x480@30fps hasta 1280x960@12fps.
- Cámara de profundidad: resolución desde 320x240@30fps hasta 640x480@30fps.
- La cámara de profundidad es capaz de medir distancias entre un rango de aproximadamente 80cm a 4m.
- Un campo de visión de 43º Vertical y 57º Horizontal.

La cámara RGB tiene características similares a una *Webcam* común, por lo que no se darán detalles sobre su funcionamiento.

La cámara de profundidad es una cámara sensible al infrarrojo, que funciona junto a un proyector IR que emite un patrón de puntos. La cámara captura dicho patrón y en función de la distorsión de estos puntos es capaz de determinar la distancia a la que se encuentra cada pixel de la imagen.

El principal problema que presenta la cámara de profundidad es la limitada resolución y velocidad de refresco (se requiere un PC de gama alta para poder capturar y procesar imágenes a 30fps). También presenta ciertos problemas para capturar imágenes con precisión, las imágenes de profundidad pueden presentar pequeños artefactos, unas líneas verticales que aparecen aleatoriamente, y aunque no influyen para ciertas tareas como *Skeleton Tracking* y otros tipos de seguimientos de objetos, si que son significativas para tareas de mayor precisión como las que se pretenden implementar en este proyecto. Estos problemas y algunos más, se tratan en el artículo [28] y en el propio *FAQ* del SDK oficial [29].

Pese a las nombradas limitaciones, existen artículos [26] [15] [17] que demuestran que se puede utilizar para tareas como seguimiento de dedos y otras tareas de detección de objetos.

Capítulo 3

Análisis

A continuación se describen los requisitos de nuestra aplicación para alcanzar los objetivos mencionados en el primer capítulo.

Comenzaré definiendo la lista de características y limitaciones, seguido por los requisitos de la aplicación, y finalizando por la descripción de los componentes del proyecto y el flujo de información a través de ellos.

3.1. Descripción de la aplicación: características y limitaciones

En esta sección se definen brevemente las características del sistema, que serán especificadas con mayor detalle en el apartado de Requisitos 3.2. Estas características pretenden cumplir los objetivos descritos al comienzo de esta memoria en la Sección 1.2. También se describirán algunas limitaciones del sistema.

3.1.1. Características

La aplicación encargada de gestionar el escritorio digital deberá:

1. Mostrar al usuario los documentos.
2. Mostrar al usuario 3 iconos (capturar, enviar a la tablet, e imprimir).
3. Para realizar la captura de un documento se mostrará el área de captura, y se esperará unos segundos antes de realizarla (el usuario puede cancelarla en este momento).

4. Comunicar el módulo de captura con la aplicación de reconocimiento de texto de forma transparente al usuario.
5. Los documentos deberán poder trasladarse, rotar sobre su plano (como si estuviesen apoyados en una mesa) y ampliarse.
6. El sistema debe ser capaz de reconocer los siguientes gestos multitáctiles:
   a) **Apuntar** al tocar un objeto o botón con un dedo.
   b) **Trasladar**, tras apuntar, mover el dedo sin dejar de hacer contacto con la mesa.
   c) **Rotar y ampliar**, tras apuntar, al tocar con otro dedo y mover este segundo dedo.
7. El sistema debe ser capaz de comunicarse con el resto de módulos.
8. El sistema debe permitir archivar los archivos en carpetas.
9. El sistema se debe poder utilizar en una superficie grande como una mesa, en la que deben poder coexistir documentos físicos con documentos digitales.

3.1.2. Limitaciones

Se pretende realizar inicialmente un prototipo funcional que tendrá las siguientes limitaciones:

1. No se diseñarán animaciones de transición. Por ejemplo, al archivar un documento en una carpeta este desaparece instantáneamente del escritorio, y lo mismo sucede al extraerlo.
2. El gestor de archivos que permite archivar o extraer documentos de carpetas simplemente realizará esa acción. No se incluirán opciones avanzadas como mover entre carpetas, copiar, etc.
3. En la medida de lo posible se intentará mostrar el estado del procesado de texto, pero ya que esto depende de la información que pueda proporcionar el correspondiente módulo realizado por otro miembro del equipo, esta puede ser escasa.
4. Dado que se utilizará una superficie grande, la resolución de la imagen del escritorio será limitada, por ello, el escritorio digital simplemente se presenta como una forma rápida y sencilla de mostrar información. No se profundizará en que esta información sea editable en el escritorio (para ello está la tableta digitalizadora).

Estas limitaciones podrán ser resueltas en futuras versiones del sistema.

3.2. Lista de requisitos de la aplicación

A continuación se muestran los requisitos para cada una de las características mencionadas en la Sección 3.1.1.

1. **Mostrar al usuario los documentos:**
   - *a*) Se mostrará como un rectángulo sobre el que se pega una textura con la imagen del documento.
   - *b*) Se utilizará una vista ortogonal para mostrar los documentos, que se presentarán en un plano frente a la cámara.
   - *c*) Los documentos se preprocesan tras la captura para eliminar márgenes, etc. y dado que este proceso es casi instantáneo, se mostrará directamente el documento preprocesado.
   - *d*) Los documentos podrán tener distintas relaciones de aspecto según el tamaño de imagen que resulte de la imagen preprocesada.
   - *e*) Se indicará si el documento está en proceso de reconocimiento o ya reconocido.
2. **Mostrar al usuario 3 iconos (capturar, enviar a la tableta, e imprimir):**
   - *a*) Se presenta al usuario como una interfaz de usuario simple.
   - *b*) Estarán situados en la parte superior derecha de la pantalla, en una posición y tamaño fijos.
   - *c*) Se mostrarán siempre por encima del resto de documentos, ocluyéndolos.
   - *d*) El botón de capturar funcionará como un botón normal (haciendo click y soltando sobre él, ejecuta una acción).
   - *e*) *Enviar a la tablet* e *imprimir* actúan como iconos "drag and drop", se arrastra y suelta un documento sobre ellos.
   - *f*) El documento arrastrado sobre uno de estos dos iconos deberá volver a su ubicación inicial tras realizar la acción.
3. **Al capturar, mostrar el área de captura, y esperar unos segundos antes de realizarla (el usuario puede cancelarla en este momento):**
   - *a*) Tras hacer click sobre el botón de captura se mostrará el área de captura (el borde de un rectángulo).
   - *b*) Se mostrará en el centro una cuenta atrás para la captura.
   - *c*) El usuario podrá cancelarla pulsando de nuevo el botón de captura.

d) Al finalizar la cuenta atrás, desaparecerá dejando que la cámara capture correctamente el documento.

4. **Comunicar el módulo de captura con la aplicación de reconocimiento de texto de forma transparente al usuario:**

   a) Cuando el usuario pulse el botón de captura, y tras la cuenta atrás mencionada, se lanzará la orden de captura que tomará rápidamente una imagen de alta resolución del documento situado bajo el área de captura.

   b) Durante el proceso de captura, debe lanzarse seguida y automáticamente el preprocesado y posterior reconocimiento de texto.

5. **Los documentos deberán poder trasladarse, rotar sobre su plano (como si estuviesen apoyados en una mesa) y ampliar:**

   a) Deberán poder realizar dichas acciones cuando el usuario lo indique mediante los gestos que se describen a continuación.

   b) Si un documento ocluye a otro durante su realización, mostrar por encima el que se está manipulando.

   c) Al ampliar no se deformará la relación de aspecto del documento.

6. **El sistema debe ser capaz de reconocer gestos multitáctiles:**

   a) Debe actuar en consecuencia, realizando las acciones indicadas en las características: Apuntar, Trasladar, Rotar y Ampliar.

   b) Recibe la ubicación de estos de el módulo de reconocimiento.

   c) Este módulo de reconocimiento debe ser poco intrusivo para el usuario y ser capaz de captar sus movimientos con precisión y rapidez.

7. **El sistema debe ser capaz de comunicarse con el resto de módulos:**

   a) Recibe acciones del módulo de detección multitáctil.

   b) Envía a la cámara la orden de tomar fotografía, y recibe dicha fotografía.

   c) Envía a la impresora la orden de imprimir un documento.

   d) Envía a la tableta la orden de editar un documento.

   e) Envía la imagen recibida de la cámara, al módulo de procesamiento. Y recibe de él, el estado del procesamiento, y el texto una vez procesado.

8. **El sistema debe permitir archivar los archivos en carpetas:**

   a) Se mostrarán un número determinado de carpetas.

b) Cualquier documento arrastrado sobre una de ellas desaparece de la vista del escritorio y se almacenará en dicha carpeta.

c) Al hacer click sobre una carpeta se mostrará una lista con los documentos que contiene. Al hacer click sobre uno de estos volverá al escritorio.

9. **El sistema se debe poder utilizar en una superficie grande como una mesa, en la que puedan coexistir documentos físicos con documentos digitales:**

   a) Se diseñará la interfaz de usuario (tamaño de botones, carpetas y documentos) teniendo en cuenta esta característica. El fondo de la aplicación también debe ser de un color que no moleste al usuario al interferir con los documentos físicos.

   b) Se necesita un medio que permita mostrar información sobre una gran superficie como esta.

   c) Se requiere un medio que soporte la coexistencia de documentos físicos (sobre los que el usuario normalmente escribirá y por tanto debe ser resistente).

3.3. Componentes del sistema

Comenzaré describiendo el sistema completo, que consta de varios componentes: un módulo encargado del reconocimiento de texto, un módulo encargado de reconocer los gestos, un módulo encargado de la interacción con el usuario, un módulo encargado de capturar los documentos y finalmente uno encargado de imprimirlos.

Se pueden observar los módulos y la relación entre ellos y el resto de componentes en la Figura 3.1.

El módulo de procesado de documentos, es el encargado de, dado un determinado documento que el usuario habrá capturado previamente, analizarlo y procesarlo interactivamente utilizando la tableta digital. Realiza las funciones de *preproceso*, *reconocimiento de texto manuscrito* y *edición en la tableta*.

El módulo de reconocimiento de dedos, es el encargado de reconocer la interacción por parte del usuario.

El módulo de captura, se encarga de procesar la solicitud de tomar una fotografía, indicar a la cámara que tome una fotografía y enviarla al sistema.

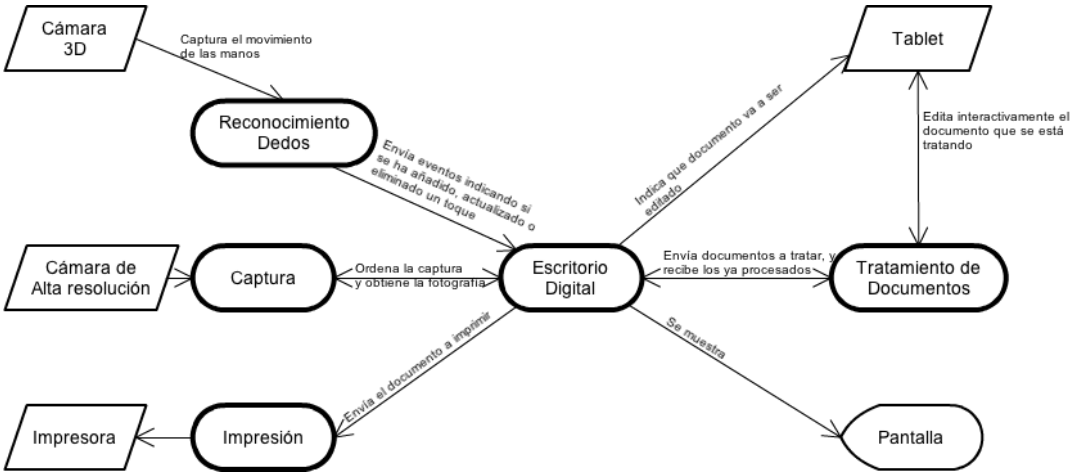


FIGURA 3.1: Módulos que componen el sistema

El módulo de impresión, permite imprimir el documento. Para obtener la mayor calidad posible, debe imprimir el texto procesado, si se ha completado por el módulo de tratamiento de documentos, o la imagen capturada en caso contrario.

El módulo Escritorio Digital se encarga de mostrar la interfaz de usuario por pantalla, permite al usuario manipular los documentos, interaccionar con el sistema y conecta el resto de módulos entre si.

De estos cinco módulos, el encargado del procesado de documentos será realizado por otros miembros del equipo del proyecto multidisciplinar. El módulo de reconocimiento de dedos también será implementado por otros miembros del equipo, aunque colaboraré en su desarrollo ya que durante la fase de búsqueda de información realicé ciertas pruebas en ese área. El resto de módulos (captura, impresión y Escritorio Digital) serán los que implementaré en esta tesina.

3.4. Flujo de trabajo

La Figura 3.2 muestra el flujo que puede seguir la información dentro del sistema. El diagrama parte de la situación en la que el sistema se encuentra funcionando y muestra el escritorio. El usuario inicia la interacción con el escritorio utilizando las manos. El módulo de reconocimiento de dedos se encarga de detectar el movimiento de las manos y enviar la información al Escritorio Digital.

El módulo del Escritorio Digital analiza la acción que el usuario ha realizado, y decide la acción a realizar. Puede, por ejemplo, solicitar al módulo de Captura que tome una fotografía (que será procesada por el módulo de procesado), manipular un documento

(cambiando su posición, tamaño y rotación), imprimir un documento o enviarlo a la tableta.

Finalmente el módulo del Escritorio Digital debe actualizar la imagen que el usuario verá.

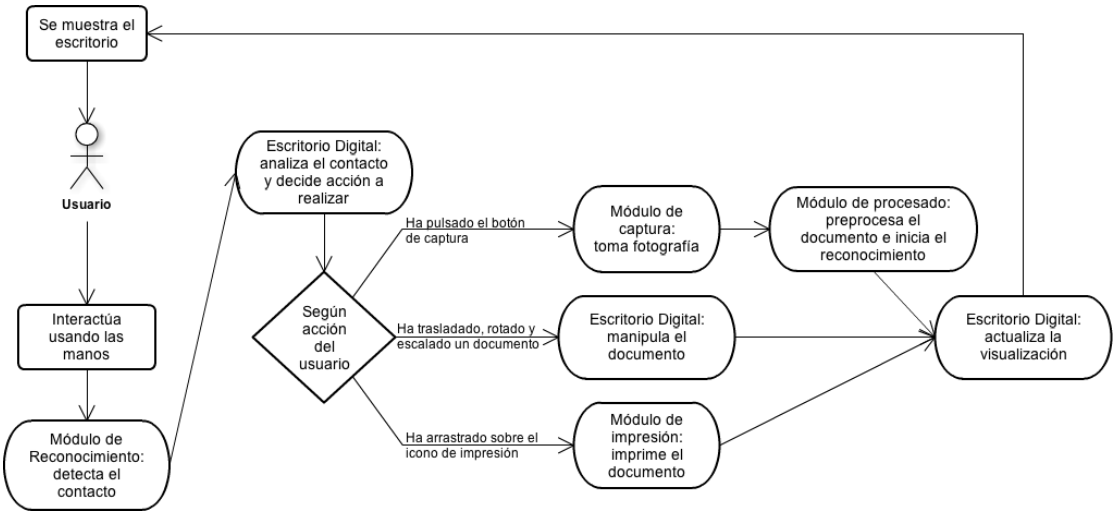


Figura 3.2: Flujo de trabajo

Capítulo 4

Diseño

4.1. Descripción general

En este apartado comenzaré describiendo la configuración física del sistema ideado, junto con las tecnologías y librerías utilizadas para ello. Posteriormente se presenta un análisis de casos de uso, seguido de un diseño de la interfaz de usuario y el diseño de los gestos utilizados para la interacción con el sistema. Por último, se presentarán unos diagramas de actividad, que muestran el flujo de la información dentro de la aplicación, y unos diagramas de clases.

4.1.1. Librerías, toolkits, hardware, tecnologías, y componentes utilizados

En este apartado se describe cómo fue el proceso mediante el cuál se decidió qué tecnología se utilizaría para desarrollar el proyecto, está separado por categorías y dentro de cada una se explican inicialmente las posibles soluciones que se barajaron, probaron, descartaron o finalmente formaron parte de la solución que se ha utilizado.

Uno de los primeros pasos fue pensar **cómo iba a ser el sistema físico** que el usuario utilizaría. Como se ha explicado anteriormente, este debe cumplir ciertos requisitos (Sección 3.2 punto 9), uno de los principales es que permita coexistencia entre documentos físicos y digitales, debe poder usarse como una mesa de escritorio normal y corriente. Esto restringe bastante inicialmente las distintas posibilidades:

- No se puede utilizar por tanto una pantalla LCD para mostrar información, principalmente por su limitado tamaño, además se calientan y no son muy resistentes a la presión (esto último podría solucionarse situando algún cristal sobre la pantalla).

- Un sistema de retroproyección, situado bajo una mesa podría ser una solución, pero dado el limitado espacio, y que el usuario va a introducir las piernas bajo ella, el sistema resultante ocuparía demasiado espacio. Se puede observar un sistema similar en el artículo *Benddesk* [10], y se puede observar como para una superficie plana horizontal no muy grande (el usuario llega con los brazos a la pantalla vertical) se requiere una distancia considerable para que las piernas no ocluyan el proyector inferior.

- Queda la siguiente **solución**, que consiste en **utilizar un proyector situado sobre la mesa**, parecido a lo que se presenta en [7] [8]. Este tipo de sistema presenta también problemas, ya que las manos ocluyen la imagen proyectada, pero para el prototipo inicial del proyecto es suficiente.

Respecto al reconocimiento multitáctil (Sección 3.2 punto 6), y dadas las restricciones existentes:

- Conseguir un panel táctil capacitivo que se ajuste al tamaño del escritorio, es prácticamente imposible, el coste de estos paneles, si se requieren de gran tamaño, suele ser alto, además tendría que ser resistente a la presión. Por lo tanto queda la opción de utilizar una cámara y visión por computador.

- De entre los distintos tipos de realizar la captura, hay que descartar aquellos que la realizan bajo la superficie, pues presentan los mismos inconvenientes que la retroproyección, ocupan la parte inferior de la mesa e impedirían usarla como mesa de trabajo.

- De entre los que quedan, se puede utilizar en el proyecto o una **cámara de profundidad** y un detector como el presentado en [26], o una **cámara y un proyector de infrarojos** como el utilizado en [7]. Cualquiera de los dos es igual de válido en un principio, pues presentan los mismos inconvenientes y ventajas.

- Intentamos montar ambos sistemas y finalmente decidimos utilizar *Kinect*, ya que el compuesto por el *proyector de infrarrojos y cámara* no mostraba una imagen utilizable ya que la luz del foco de infrarrojos estaba muy concentrada y no era posible distribuirla uniformemente por toda la mesa.

- Probé a realizar un detector como el propuesto en [26], pero dado que ese tipo de detector sólo captura los contactos de cualquier objeto con la mesa, no nos servía para el proyecto (ya que habrán más elementos sobre la mesa), se descartó, pero siguiendo la idea propuesta, otro miembro del equipo realizó un detector capaz de distinguir entre una mano con un dedo extendido y otros elementos.

Es necesario pensar en el software que se encargará de manejarlo (Sección 3.2 puntos 1,2,3,5,8), construir todo partiendo desde cero sería demasiado costoso, así que se utilizarán diversas tecnologías, librerías y toolkits existentes para facilitar el trabajo. Para el escritorio digital:

- Se necesita algún tipo de motor gráfico que permita renderizar los distintos componentes, y aplicar las transformaciones necesarias sobre ellos.
- Inicialmente se presentó la posibilidad de utilizar un sistema de ventanas y un escritorio común como el que incluyen la gran mayoría de sistemas operativos. Esta opción se descartó inicialmente, ya que en los escritorios tradicionales, existe una fuerte restricción y es que las ventanas no pueden rotar, sus bordes deben ser paralelos a los del escritorio (aunque hoy en día existen extensiones que permiten modificar las ventanas, incluso deformándolas, a la hora de manejar su contenido estas no pueden estar rotadas).
- Siguiendo con la idea de un escritorio con ventanas, se analizó la posibilidad de utilizar un nuevo gestor de ventanas llamado *Wayland* [30], disponible para Linux, y que sustituyendo al clásico protocolo *X Window System* presenta grandes ventajas, como el poder utilizar ventanas sea cual sea su orientación. El problema es que se encuentra en una versión muy inicial, muy inestable, y tras intentar hacerlo funcionar repetidas veces, lo descarté.
- En la línea de lo anterior, otra opción para poder manipular ventanas que han sido transformadas, consiste en renderizarlas en un servidor VNC y utilizar la imagen que este devuelve como textura en un motor gráfico. Se puede observar el funcionamiento de un sistema similar en el artículo [18].

  En el experimento probé a utilizar OSG, modificando el cliente de VNC que incluye para que pudiese tratar varias conexiones, y aunque el sistema funcionaba, presentaba muchos problemas. El principal problema se daba a la hora de tratar oclusiones, cada servidor VNC sólo debía renderizar una ventana para poder tratarla individualmente, pero ese no es el funcionamiento tradicional de las aplicaciones, que generalmente abren nuevas ventanas. Otro gran problema es el redimensionado de las ventanas, imposible cambiar la resolución del servidor VNC al vuelo. Además, al añadir los distintos servidores VNC para mostrarlos con el cliente de OSG, algunas veces no cargaba la imagen que el servidor enviaba (aparecía una textura negra en lugar del escritorio), y por ello lo descarté.
- Queda por lo tanto, la solución utilizada, que consiste en utilizar un motor gráfico, y renderizar con él los distintos elementos, de esta forma, aunque el sistema resultante será menos flexible que los planteados anteriormente, se ajustará a lo que

necesitamos, y simplificará la tarea del usuario al no mostrarle nada más que los elementos de la aplicación. Como motor gráfico se eligió *OpenSceneGraph* [31] ya que se ha estudiado en algunas asignaturas del máster, código libre, y funciona en gran cantidad de sistemas operativos.

Respecto ala captura de los documentos (Sección 3.2 punto 4), ha de realizarse sobre el escritorio virtual, y hay que realizarlo de una forma rápida:

- La mejor opción es realizar la captura con una cámara, ya que un escaner ocuparía mucho espacio sobre la mesa y sería más lento.
- Esta cámara irá situada sobre la zona de captura, en una ubicación fija y enfocada a la zona de captura.
- En un principio se usó una cámara *Canon IXUS 860 IS*, que utilizando el *firmware CHDK* [32] y la aplicación *CHDKptp* permite tomar fotos cuando se le indica desde el PC.
- La solución de captura implementada finalmente ha sido utilizando la aplicación *gPhoto*, que permite realizar capturas desde un ordenador con ciertos modelos de cámara [33], entre ellas la *Canon EOS 1100D* [34], que es la seleccionada, junto con el objetivo *Canon EF-S 18-55mm f/3.5-5.6 IS II* [35].

El módulo que se encarga de la impresión de documentos utiliza una impresora normal, que podrá situarse junto a la mesa, y se imprimirá utilizando las funciones o comandos del sistema operativo. Hay que tener en cuenta que existen dos tipos de documentos a imprimir, imágenes si el documento no se ha procesado y texto plano si lo está. Para la impresión de texto plano con un formato correcto y la correcta separación en líneas se podría procesar con LaTeX, definiendo inicialmente un formato de documento, e imprimiendo el resultado.

El módulo encargado de la comunicación con la tableta simplemente ha de ser capaz de enviar la orden de abrir una determinada página del navegador, ya que la aplicación de edición existente en el módulo de procesado funciona de esta forma.

El sistema resultante será por tanto algo similar a lo que se describe en el diagrama de la Figura 4.1. En él, falta incluir la impresora, pero se omite, ya que como he comentado anteriormente, puede situarse junto a la mesa, o incluso utilizar una de las impresoras en red disponibles en el edificio, etc. En el diagrama no se incluyen distancias, pues estas dependen del tamaño de la mesa, y aunque se muestran todos los componentes a la misma altura, cada uno debe estar situado a una cierta distancia de la mesa según sus características.

Si utilizamos Kinect, el campo de visión horizontal es 58º [36], por lo que podemos calcular la altura a la que hay que colocar la cámara:

$$\tan\left(\frac{campo\ de\ visión}{2}\right) = \frac{\frac{ancho\ mesa}{2}}{altura} \tag{4.1}$$

Dado que la mesa que utilizamos en el montaje en el laboratorio mide 120cm:

$$altura = \frac{\frac{ancho\ mesa}{2}}{\tan\left(\frac{58°}{2}\right)} = \frac{\frac{120}{2}}{\tan\left(29°\right)} = \mathbf{108{,}24\ cm} \tag{4.2}$$

El objetivo de la cámara, con las especificaciones disponibles en [35], muestra un campo de visión horizontal des 64°20′ a 23°20′ según la distancia focal utilizada. A mayor distancia focal, presenta un menor ángulo de visión, pero también una menor distorsión de perspectiva, por ello se utilizará la mayor distancia focal posible.

$$\tan\left(\frac{campo\ de\ visión}{2}\right) = \frac{\frac{ancho\ del\ documento}{2}}{altura} \tag{4.3}$$

Los documentos manuscritos que se utilizan generalmente tienen un tamaño máximo A4 (29,7cm · 21cm), de esta forma:

$$altura = \frac{\frac{ancho\ del\ documento}{2}}{\tan\left(\frac{23°20'}{2}\right)} = \frac{\frac{29{,}7}{2}}{\tan\left(11°40'\right)} = \mathbf{71{,}91\ cm} \tag{4.4}$$

Las características del proyector suelen indicar la *distancia de proyección*, así que no se necesita calcular.

Existen multitud de proyectores con diferentes características, y dado que la distancia y área de proyección dependen de ellas, podemos recurrir a una base de datos web como la disponible en [37], para encontrar uno que se ajuste a las siguientes características:

- Distancia de proyección corta (alrededor de 140cm máximo, ya que el techo se encuentra sobre los 250cm, la mesa tiene una altura de 75cm y el proyector hay que colocarlo de forma vertical apuntando hacia abajo y supongamos que mide 30cm $(250 - 75 - 30 = 145cm)$).
- Sea capaz de abarcar la mesa a esa distancia, la mesa del laboratorio mide 80cm x 120cm.

Finalmente queda unir los distintos módulos (Sección 3.2 punto 7):

- La detección multitáctil y el escritorio se comunicarán por medio de la librería *TUIO* [38] que fue diseñada como una capa de abstracción entre la aplicación y

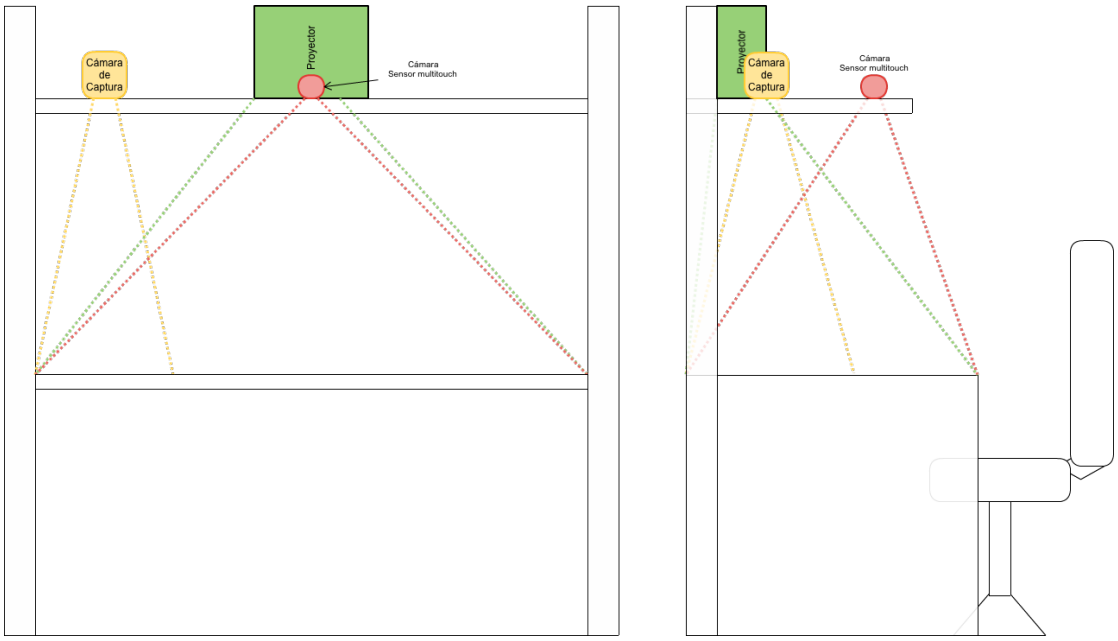


FIGURA 4.1: Diagrama del espacio de trabajo: en él se observa un esquema, alzado y perfil, del escritorio con los componentes montados, así como los campos de visión que deberán abarcar los distintos componentes.

la detección de punteros y objetos tangibles (o dedos) en el proyecto *reacTIVision* [39]. Se presenta como una interfaz en la que la aplicación encargada de recibir los datos inicializa un *TUIO Client*, y la aplicación encargada de la detección envía ciertos mensajes UDP a la aplicación cliente indicando los parámetros del toque.

- La comunicación con la cámara se realiza con una cámara que soporte la captura controlada por ordenador a través de cierto software (*CHDKptp* o *gPhoto* como se ha explicado en el apartado anterior).
- La comunicación con el módulo de tratamiento de texto se realizará, bien por algún protocolo de mensajes ideado especialmente para la ocasión, bien lanzando los *scripts* automáticamente desde la aplicación, o bien enviando los archivos a analizar a una determinada carpeta que se encargará de vigilar.
- La comunicación con la tableta simplemente consiste en indicar la url del editor, esta luego se comunica directamente con el módulo de tratamiento de texto.

4.1.2. Diagrama de Casos de uso

En el siguiente diagrama de la Figura 4.2 se puede observar rápidamente los distintos casos de uso que se pueden presentar en la aplicación. A continuación se describen con mayor detalle en los diagramas de actividad.

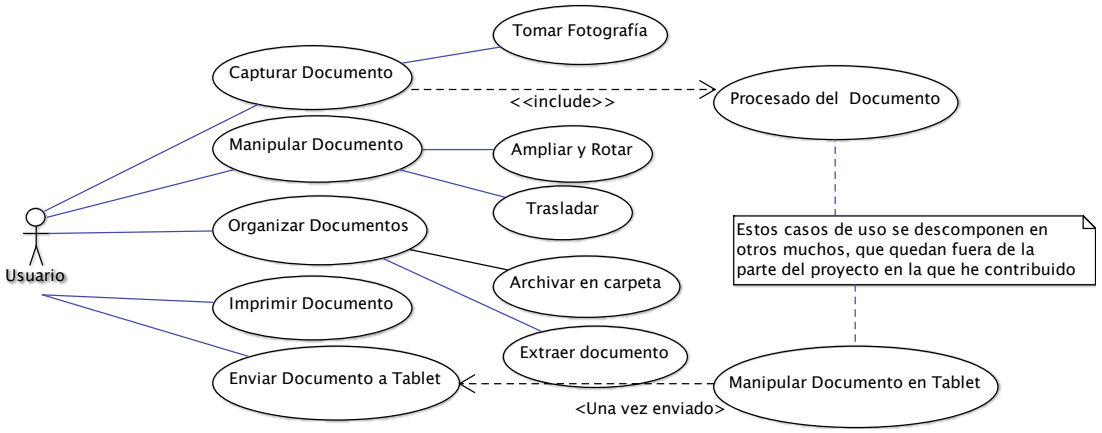


FIGURA 4.2: Diagrama de casos de uso

1. **Capturar Documento**, que se puede descomponer en dos: "Tomar fotografía" y "Procesado del Documento" (siendo este último, junto a más detalles del procesado, responsabilidad de otro miembro del equipo). El usuario sitúa un documento sobre el área de captura, toca el botón correspondiente, y tras esperar la cuenta atrás (y asegurarse de que el documento está bien encuadrado), el sistema llama al módulo de captura e indica a la cámara que tome una fotografía. Esta fotografía pasa al mencionado módulo de tratamiento de texto, donde se procesa.

2. **Manipular Documento**, considerado como un caso de uso más general, en el que se modifican las propiedades de visualización de un documento. Se descompone en dos:

   a) **Ampliar y Rotar**, donde el usuario toca y mantiene con un dedo un documento, y con un segundo dedo arrastra para ampliar y rotar.

   b) **Trasladar**, donde el usuario toca y mantiene con un dedo un documento, y mueve dicho dedo para desplazar el documento por la pantalla.

3. **Organizar Documentos**, refiriéndose al proceso de almacenado de documentos, se descompone en dos:

   a) **Archivar en carpeta**, donde el usuario arrastra (al igual que al Trasladar) sobre una carpeta el Documento, y este desaparece de la vista para pasar a almacenarse en la carpeta.

   b) **Extraer documento**, el usuario toca una carpeta, esta muestra su contenido y el usuario toca el Documento que quiera mover de nuevo al escritorio.

4. **Imprimir Documento**, el usuario arrastra (al igual que en Trasladar) el Documento sobre el icono de imprimir, se lanza la acción y el Documento vuelve a la posición original.

5. **Enviar Documento a Tableta**, el usuario arrastra (como en Trasladar) el Documento sobre el icono de la Tableta, se lanza la acción, el Documento vuelve a su posición inicial, y la tableta digitalizadora se comunica con el módulo correspondiente (desarrollado por otro miembro del equipo).

4.1.3. Interfaz de usuario

Un posible diseño sería el que se muestra en la Figura 4.3, donde se puede observar la mesa de trabajo, con unos documentos sobre ella, los botones mencionados, y algunas carpetas. El área de captura no se mostraría hasta que no se pulse el botón correspondiente, pero se indica con una línea de puntos.

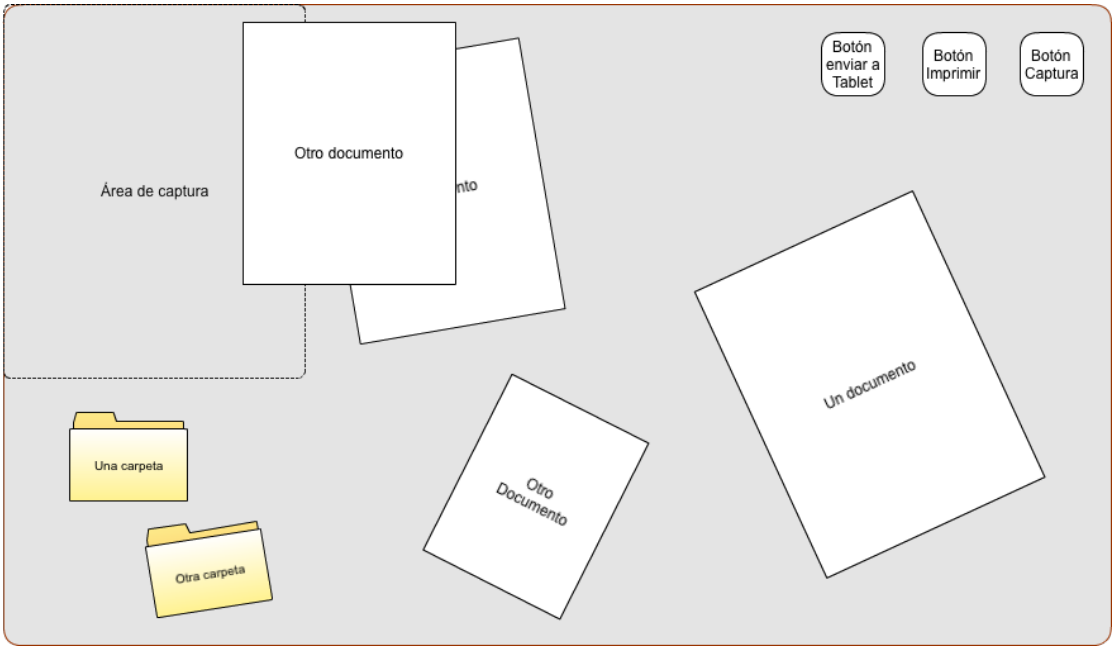


FIGURA 4.3: Vista global de la interfaz de usuario

A continuación, en la Figura 4.4, se muestra la interfaz al mostrar el área de captura, tras pulsar el botón correspondiente.

En la Figura 4.5 se muestra la interfaz de usuario mostrando el contenido de una carpeta.

No se muestran el resto de acciones de la interfaz de usuario, como arrastrar, ampliar, rotar, enviar a la tableta, e imprimir, ya que las acciones que realizan sobre la interfaz son obvias, o no modifican su apariencia. Estas acciones están explicadas a continuación en la Sección 4.1.4.

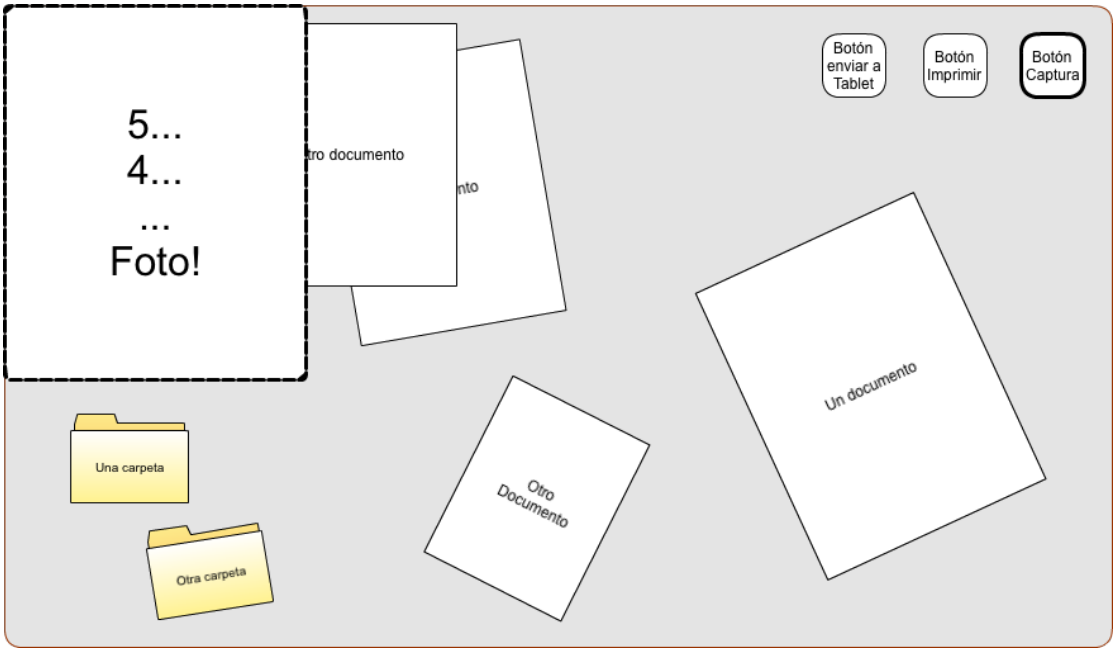


FIGURA 4.4: Interfaz de usuario durante la captura

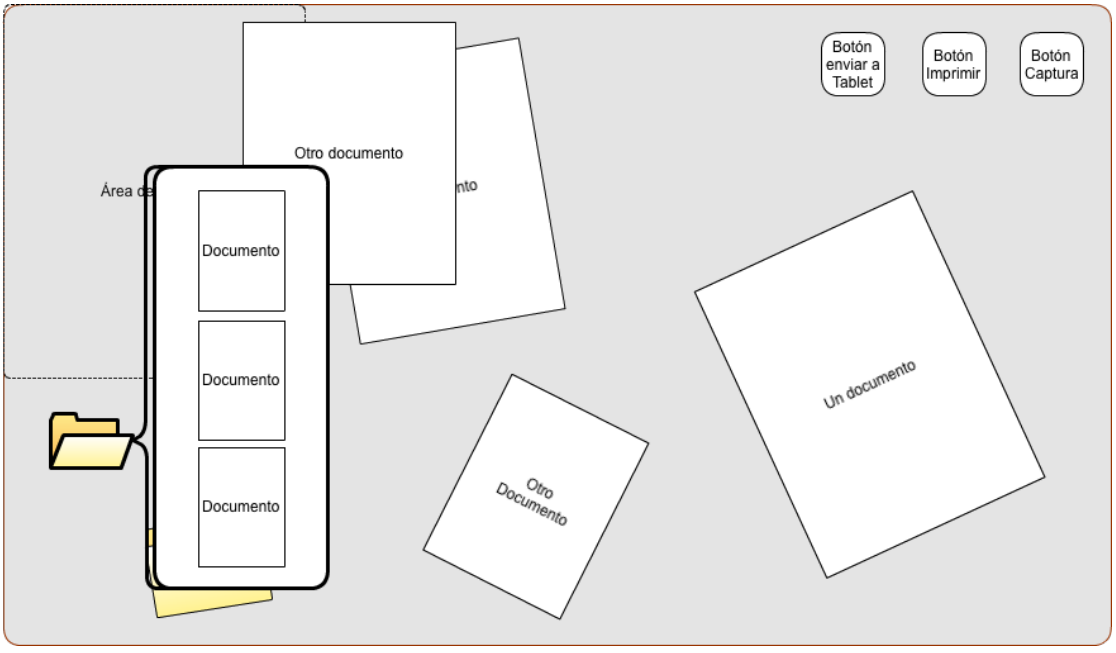


FIGURA 4.5: Interfaz de usuario mostrando el contenido de una carpeta

4.1.4. Diseño de la interacción mediante gestos

En este apartado se explican cómo se han diseñado los gestos que permiten realizar las acciones requeridas, descritas en la lista de Características, Sección 3.1.1 puntos 5 y 6, y Requisitos, Sección 3.2.

Trasladar

La Figura 4.6 muestra los pasos a realizar para trasladar un documento. En ella se puede observar como inicialmente hay que realizar el gesto de *Apuntar* un documento, y seguidamente *Trasladar* para mover el Documento. Finalmente se deja de apuntar el documento levantando el dedo y este se queda fijo en la nueva ubicación.

En el diagrama se muestra el documento arrastrado con una línea de puntos simplemente para facilitar su identificación, en la aplicación el documento se mueve a la vez que el dedo, por lo que no es necesario.

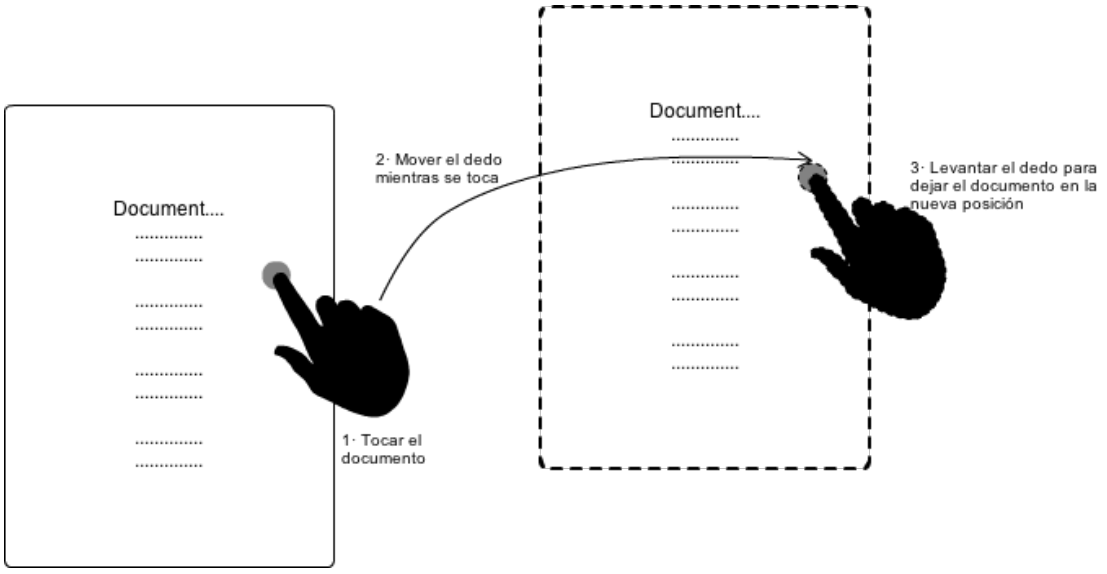


FIGURA 4.6: En el diagrama se muestran los pasos necesarios para trasladar un Documento de una ubicación a otra en la pantalla.

Rotar y ampliar

En la Figura 4.7 se muestran los gestos necesarios para ampliar y rotar un documento. Inicialmente hay que *Apuntar* el Documento que se quiere manipular, seguidamente se apunta en otro punto de la pantalla y se *Traslada* este punto para escalar y rotar el documento seleccionado anteriormente. Finalmente se deja de apuntar el documento levantando los dedos y este se queda fijo con las nuevas propiedades.

En este diagrama, al igual que en el anterior, se muestra el Documento manipulado con una línea de puntos, y además desplazado hacia la derecha. En la aplicación el Documento rota y se amplia siguiendo los movimientos del usuario.

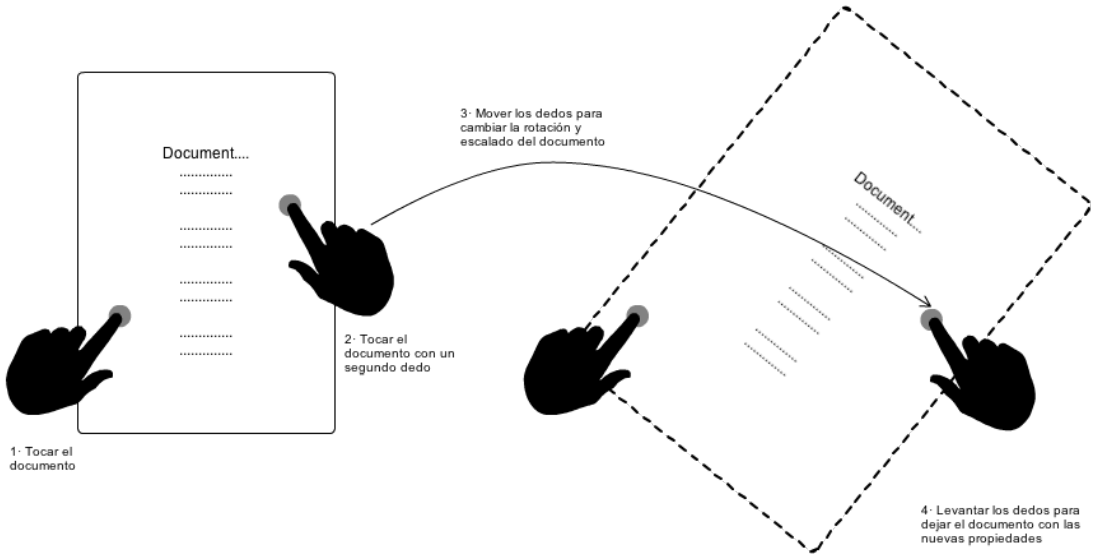


FIGURA 4.7: En el diagrama se muestran los pasos necesarios para rotar y/o ampliar un Documento.

Interacción con los botones

En la Figura 4.8 se muestra el proceso necesario para imprimir un Documento. Hay que comenzar *Apuntando* un Documento, luego se *Traslada* y se deja sobre el botón (el de Imprimir en este caso) para que se realice la acción. Tras levantar el dedo sobre un botón, el documento vuelve a su posición inicial en pantalla.

El caso de enviar un documento a la *Tablet* o una *Carpeta* sería equivalente, pero arrastrando el Documento sobre cada uno respectivamente.

Existe otro caso particular, los botones que reaccionan a *clicks*, como el de *Captura* y las propias *Carpetas*. En estos botones, el usuario debe *Apuntar* o soltar sobre ellos para realizar la acción. No se incluye un diagrama de este caso dada su simplicidad.

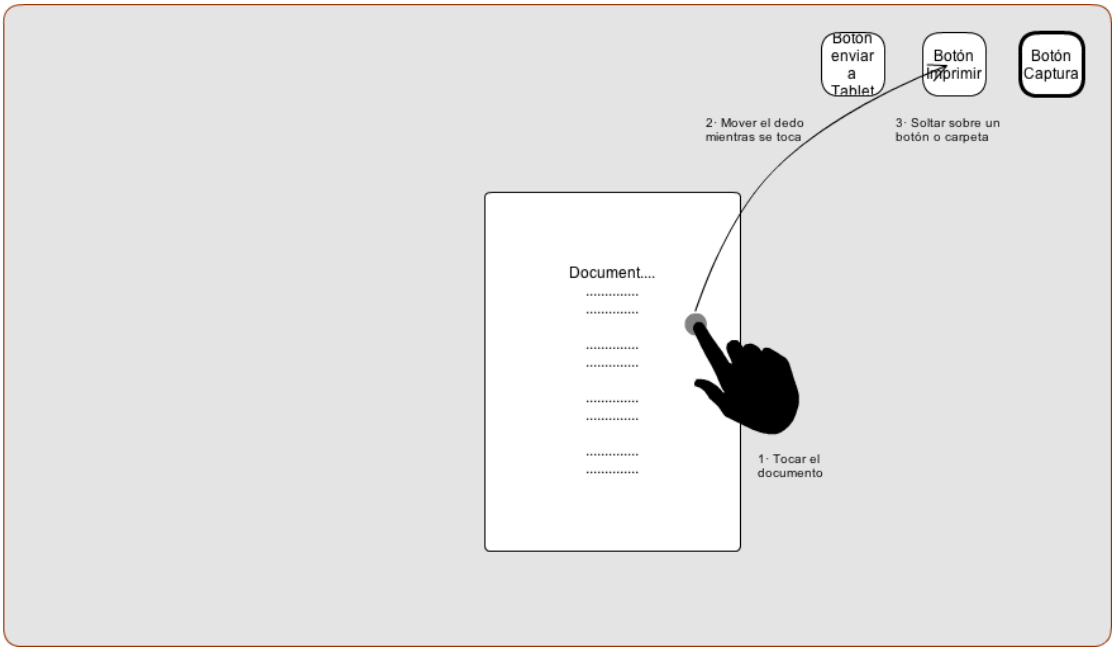


FIGURA 4.8: En el diagrama se muestran los pasos necesarios para imprimir un Documento.

4.2. Diagramas de Actividad

4.2.1. Capturar Documento

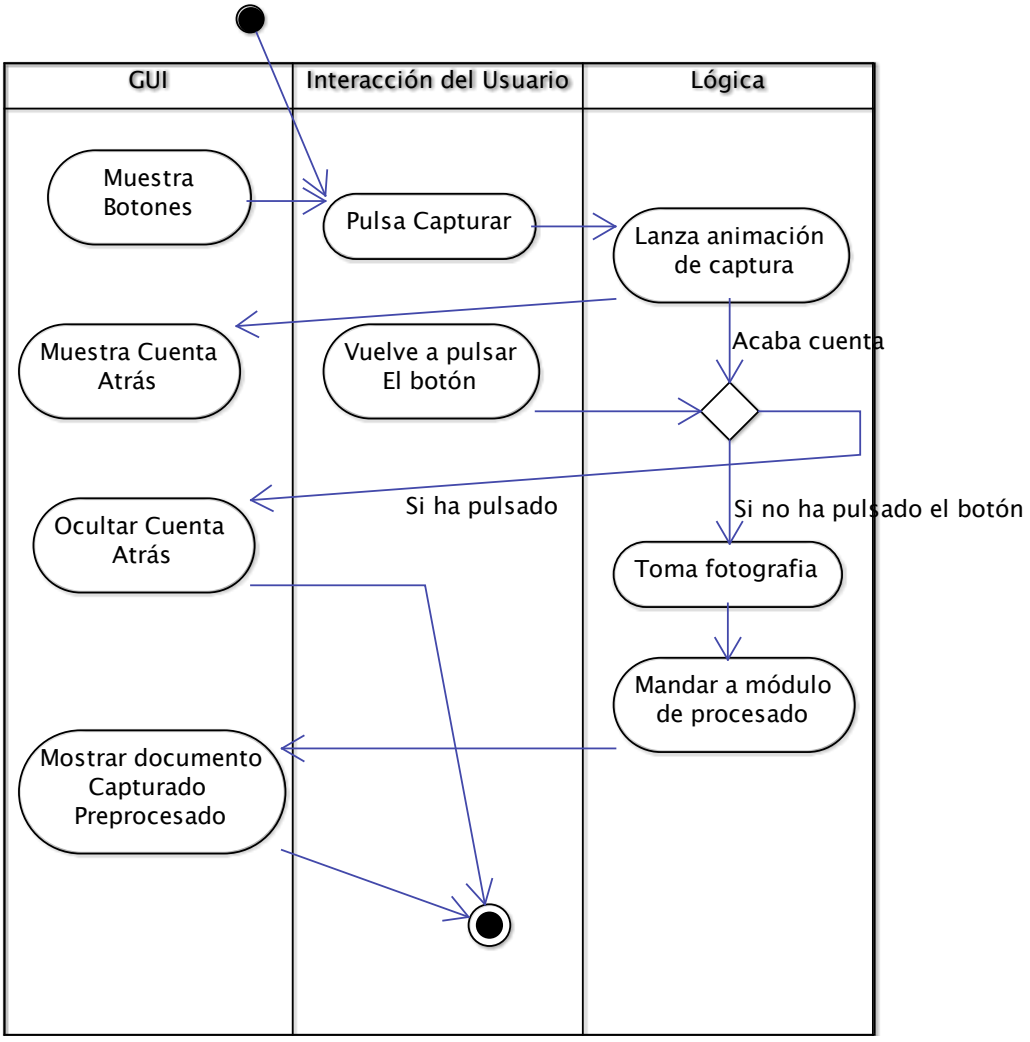


FIGURA 4.9: Diagrama de Actividad - Capturar Documento

En el diagrama de la Figura 4.9 se observa el posible flujo del caso de uso "Capturar Documento", la única alternativa que puede surgir es si durante la cuenta atrás el usuario vuelve a pulsar el botón, cancelando con ello la captura.

4.2.2. Trasladar

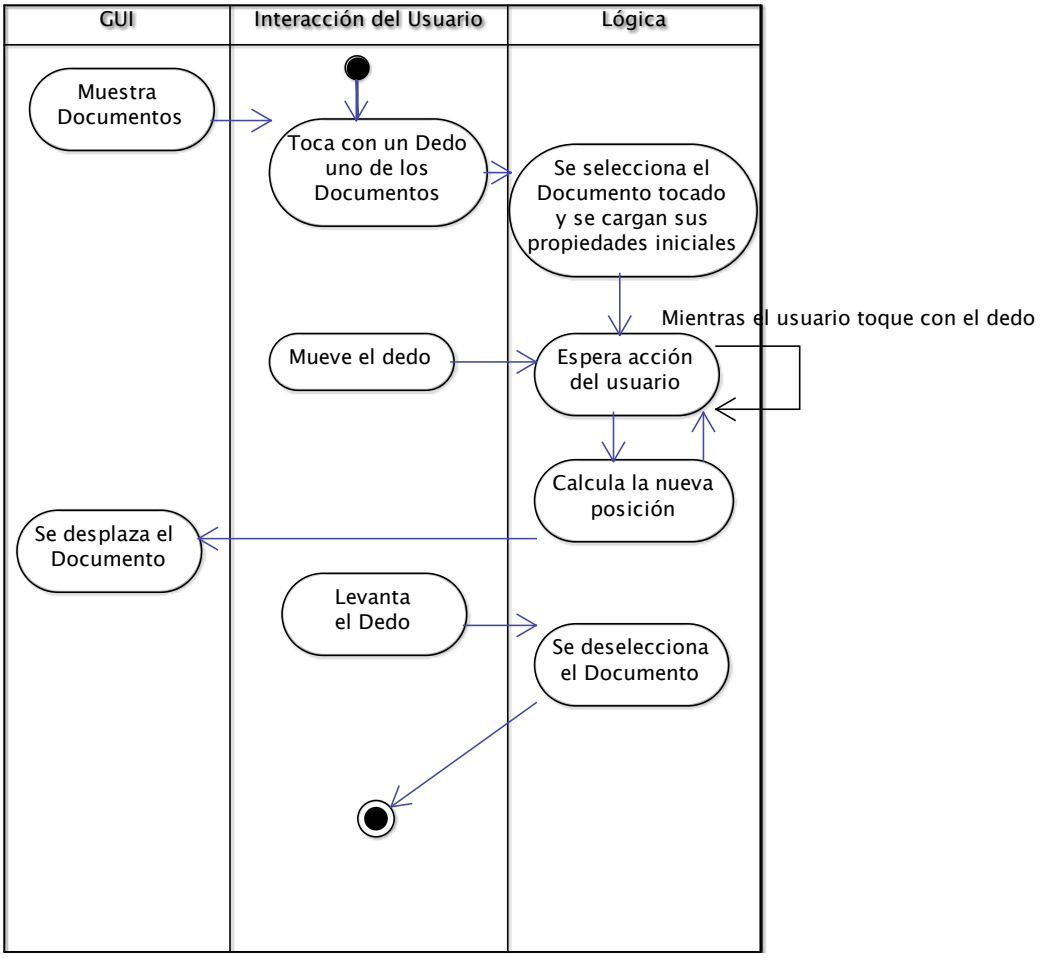


FIGURA 4.10: Diagrama de Actividad - Trasladar Documento

El diagrama mostrado en la Figura 4.10 es algo más complejo, ya que el sistema espera, calcula y mueve el objeto mientras el usuario esté tocándolo con el dedo. Al levantar el dedo se deseleccionaría el objeto finalizando esta actividad.

4.2.3. Rotar o Ampliar

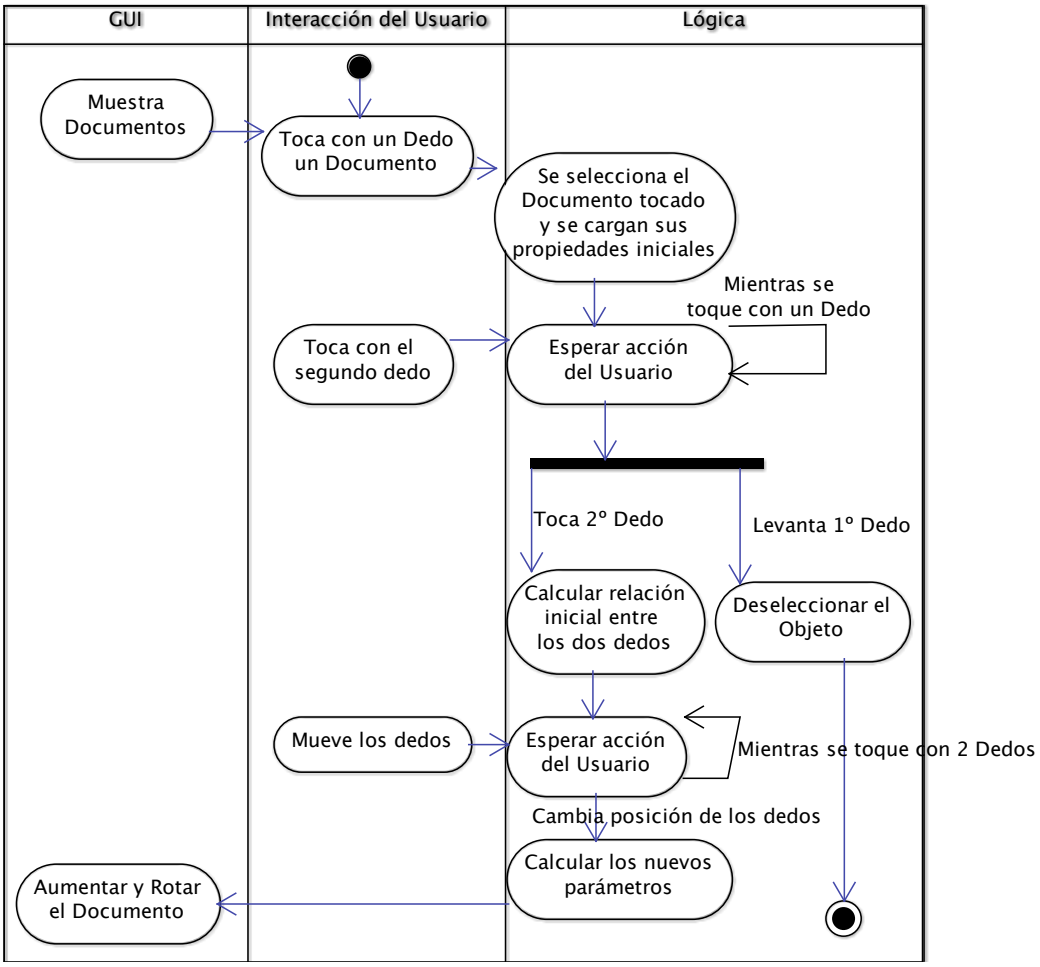


FIGURA 4.11: Diagrama de Actividad - Ampliar o rotar Documento

En el diagrama mostrado en la Figura 4.11, inicialmente se actúa como en el anterior, con la diferencia de que si mientras se está tocando con un dedo se toca con otro se pasa a este modo, en el que se calcula el escalado y rotación del objeto en base a la relación entre los dos dedos. Al soltar uno se pasaría al caso anterior y cuando no quede ningún dedo tocando se deseleccionaría el objeto finalizando esta actividad.

4.2.4. Archivar

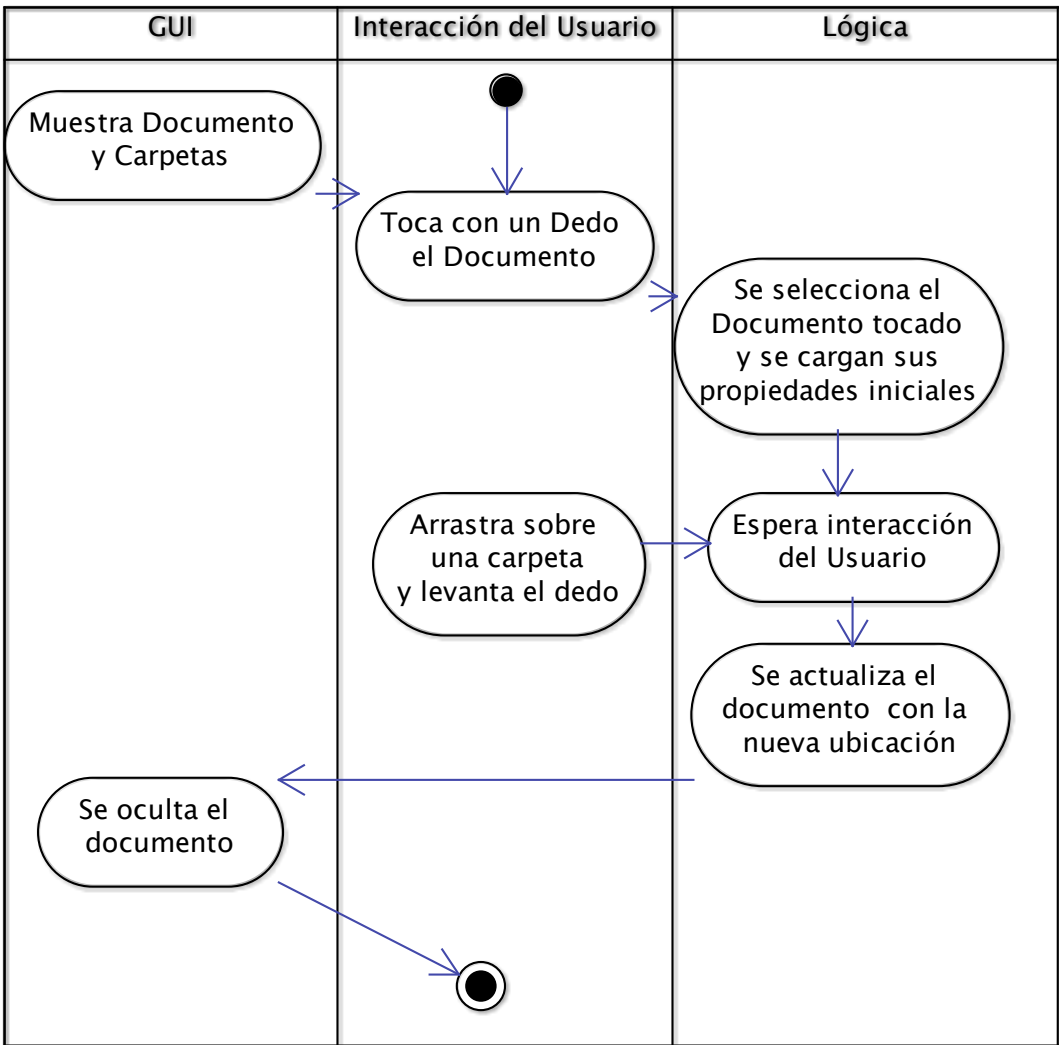


FIGURA 4.12: Diagrama de Actividad - Archivar Documento en carpeta

En la Figura 4.12 se muestra el diagrama de acciones necesarias para archivar. Es parecido al de Trasladar, y la única acción realizada es ocultar el documento y archivarlo en la carpeta.

4.2.5. Extraer documentos

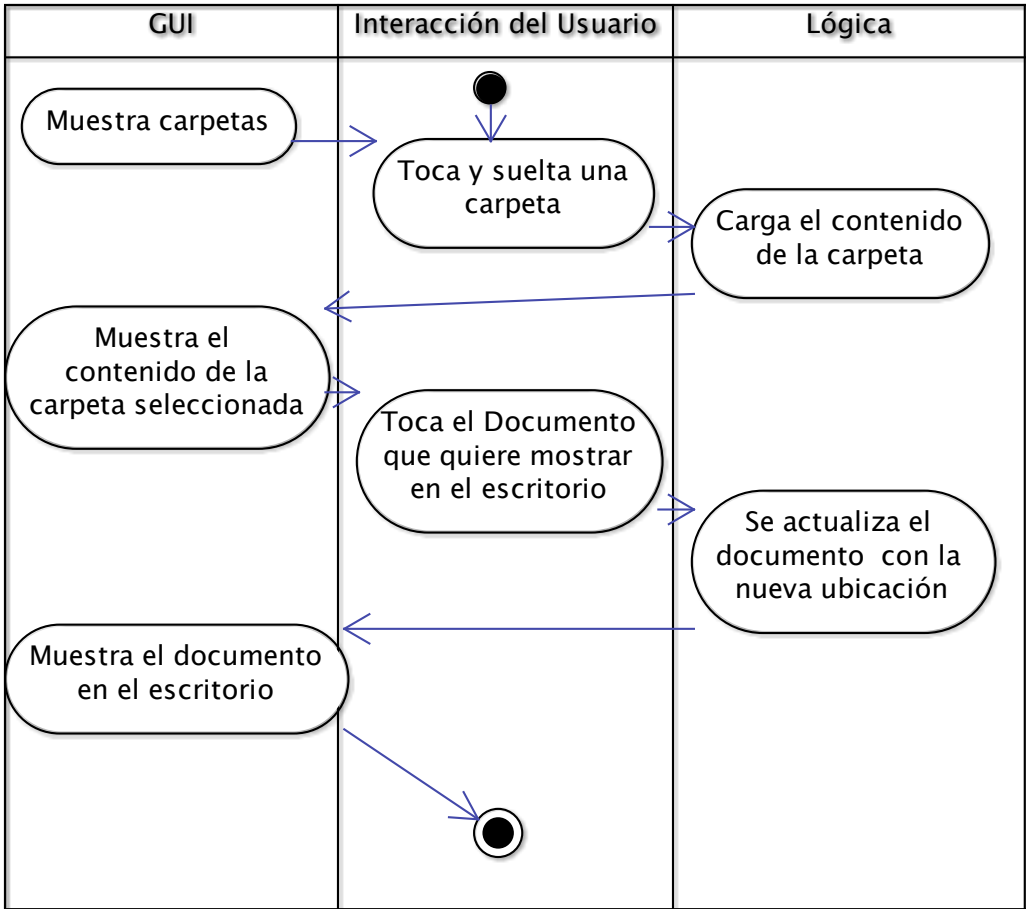


FIGURA 4.13: Diagrama de Actividad - Extraer un Documento de una carpeta

En la Figura 4.13 se muestra el diagrama de acciones necesarias para extraer un documento. Básicamente se trata de realizar el proceso contrario a archivar. Este en cambio tiene que mostrar el contenido de la carpeta, lo cual puede complicar algo su implementación.

4.2.6. Imprimir

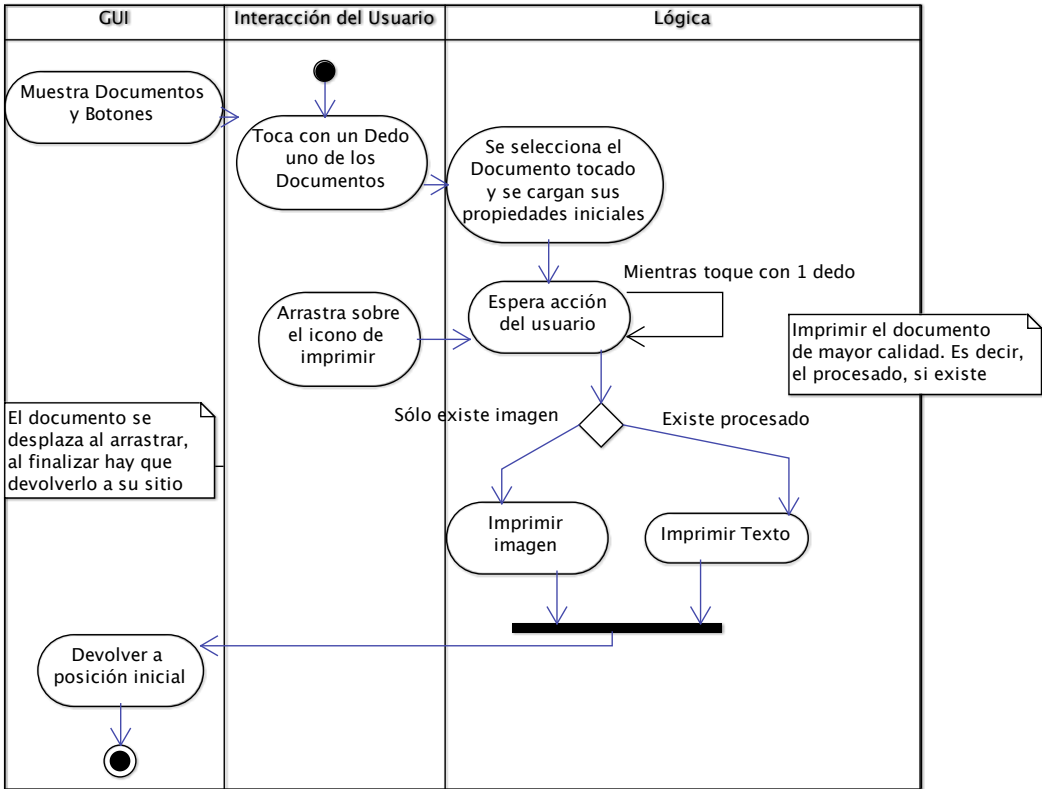


FIGURA 4.14: Diagrama de Actividad - Imprimir Documento en carpeta

La Figura 4.14 muestra el proceso de la acción Imprimir. Existe un punto de decisión en el que se selecciona qué imprimir. Teniendo en cuenta que se quiere obtener la mayor calidad posible, si se ha completado el reconocimiento del texto del documento, se imprimirá el fichero de texto que lo contiene (dado que el reconocedor proporciona un fichero de texto plano, será necesario darle formato para poder imprimirlo correctamente). En caso de que sólo exista la captura del documento, se imprimirá como si una imagen normal se tratase.

4.2.7. Enviar a la tableta digitalizadora

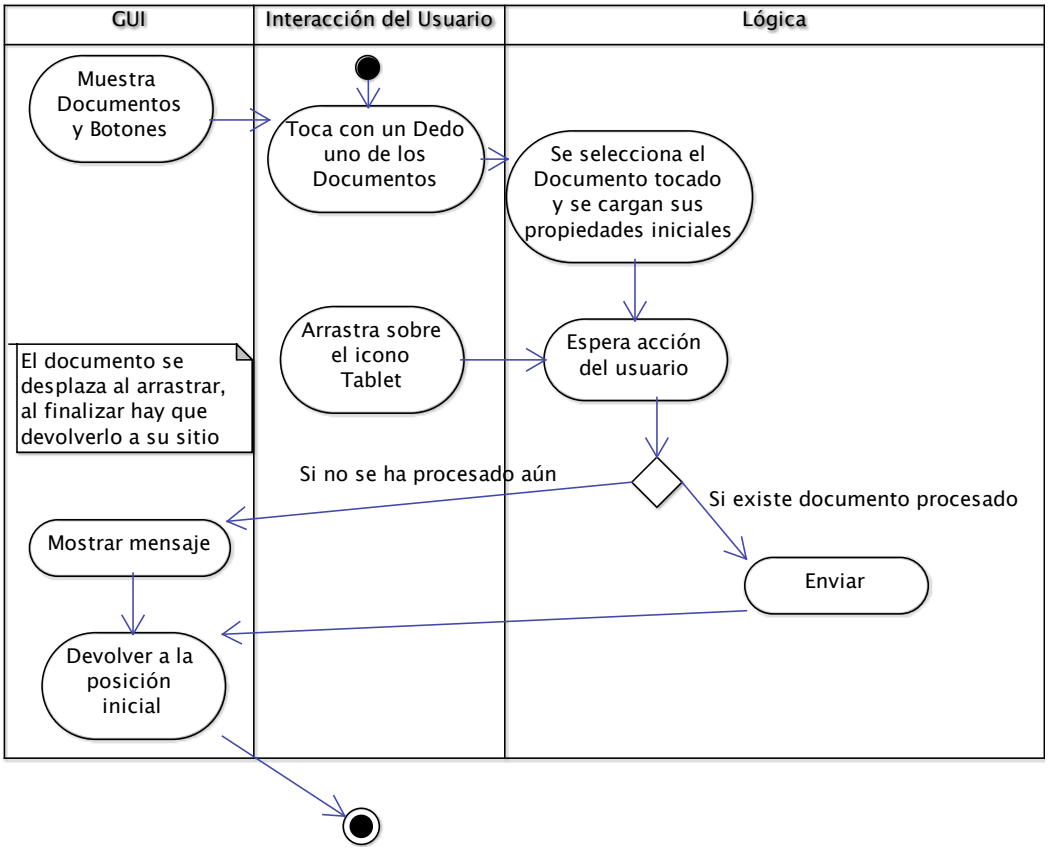


FIGURA 4.15: Diagrama de Actividad - Enviar a la tableta digitalizadora Documento en carpeta

La Figura 4.15 muestra la acción Enviar a la tableta digitalizadora. Puede darse el caso en que el documento aún no haya comenzado a procesarse, y no pueda editarse interactivamente. Por ello se mostrará un mensaje al usuario y se finalizará devolviendo el documento a su posición inicial.

4.3. Diagrama de Clases

La Figura 4.16 muestra una versión simplificada del diagrama de clases que componen la aplicación. En gris se muestran las clases de las distintas librerías de las que dependen las que he implementado.

Por simplicidad se omiten la gran mayoría de los métodos, constructores y atributos, así como otras clases auxiliares y componentes como enumeraciones, etc. que son necesarios pero no relevantes a esta escala.

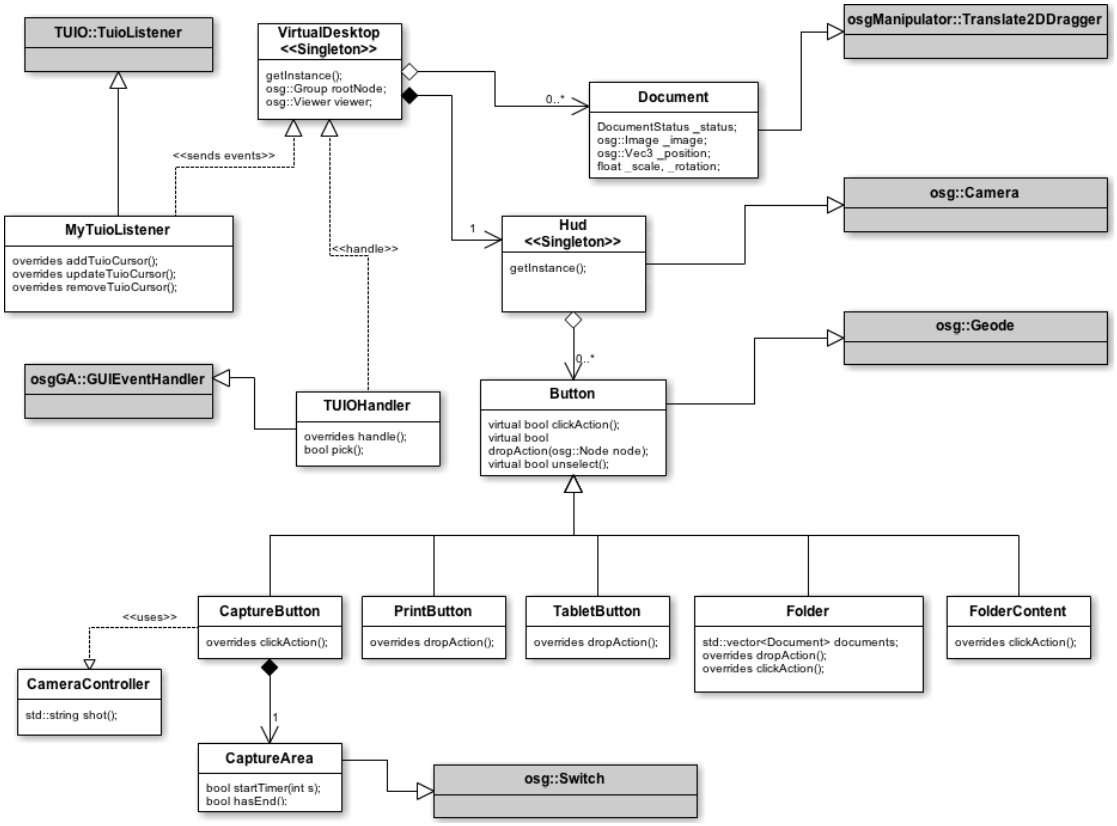


FIGURA 4.16: Diagrama de Clases, versión simplificada para explicar los componentes principales del sistema

En las siguientes páginas se detalla el comportamiento de cada una de las clases del diagrama.

4.3.1. VirtualDesktop

Es la clase que podríamos denominar "principal" del sistema. Es llamada en el método *main* y se encarga de iniciar el sistema.

Contiene un *osg::Viewer* que permite mostrar la escena definida en el nodo *osg::Group rootNode*. Es la encargada de cargar el estado anterior del sistema y guardar el actual al finalizar, e inicializar los distintos componentes del sistema.

Está implementada como un "Singleton".

4.3.2. HUD

Esta clase se encarga de mostrar la interfaz de usuario fija, que se compone de botones y carpetas. Consiste en una nueva cámara que se dibuja tras el resto de elementos del sistema, con vista ortogonal y en la que se añaden como hijos los nodos que representan los distintos elementos.

También está implementada con el patrón "Singleton". Esta clase se añade al nodo raíz de la escena comentado anteriormente y OSG se encarga de mostrarlo como corresponde.

Quizá un nombre más adecuado hubiese sido uno que sugiriese una interfaz de usuario (algo como *GUI, graphical user interface*), pero dado que se ha realizado como lo que en el mundo de los videojuegos se conoce por *HUD (Heads-up-display)* he creído conveniente mantener ese nombre.

4.3.3. Document

Esta clase se emplea para representar los Documentos que se muestran y manejan en el sistema. Hereda de la clase *osgManipulator::Translate2DDragger*, que implementa un manipulador que permite trasladar el objeto en las dos dimensiones de un determinado plano.

De esta forma, se puede mover el objeto por la pantalla y no es necesario implementar a mano las distintas conversiones que hay que hacer para trasladar un objeto desde un punto en pantalla a otro, ya que si no sería necesario convertir la posición del objeto en el mundo a un punto en pantalla, calcular el desplazamiento y volver a convertir ese desplazamiento en pantalla a un desplazamiento en el mundo (que al fin y al cabo es lo que realiza internamente).

Para que esta clase funcione, el manejador (*TUIOHandler* que se explicará a continuación) debe generar los eventos *osgGA::GUIEventAdapter::PUSH* al hacer *click*, *osgGA::GUIEventAdapter::DRAG* al mover, y *osgGA::GUIEventAdapter::RELEASE* al soltar.

Dado que la clase *osgManipulator::Translate2DDragger* hereda de *osg::MatrixTransform*, se puede aplicar también una matriz de rotación y escalado fácilmente, y de esta forma realizar todas las operaciones requeridas sobre los documentos, definidas en la Sección 3.2 punto 5.

Por último la clase contiene información al respecto del documento al que hace referencia, su estado, etc.

4.3.4. Button

Cada uno de los elementos de la interfaz de usuario que se muestra en la clase *HUD* se implementa como un *Button*. Esta clase tiene fundamentalmente tres métodos: *clickAction()*, que se ejecuta al pulsar y soltar sobre un botón; *dropAction(node)*, que se ejecuta al arrastrar otro nodo sobre el botón; y *unselect()*, que se ejecuta sobre el botón anteriormente seleccionado al hacer click fuera de él.

Todos estos métodos se deben implementar en las clases que necesiten realizar alguna de estas funciones, como pueden ser *CaptureButton*, *PrintButton*, *TabletButton*, *Folder* y *FolderContent*.

Clases que implementan botones

La clase *CaptureButton* implementa el método *clickAction()* ya que cuando esta se pulse su icono es necesario ejecutar la acción de capturar. Utiliza la clase *CameraController*, que representa el módulo de captura y se encarga de la comunicación con la cámara. Antes de tomar la fotografía se muestra el área de captura junto con una cuenta atrás, que esta implementado en la clase *CaptureArea*.

Las clases *PrintButton* y *TabletButton* implementan el método *dropAction(node)*, que se ejecutará al arrastrar y dejar un documento sobre su icono.

La clase *Folder* implementa los tres métodos mencionados, ya que debe responder tanto a clicks (mostrando los documentos), como cuando se arrastre un documento sobre su icono (archivándolo) y ocultar el contenido cuando se haga click fuera de ella. Al hacer click mostrará elementos de la clase *FolderContent*, que implementan *clickAction()* ya que al pulsarlos deben volver al escritorio.

4.3.5. MyTuioListener

Esta clase hereda de *TuioListener* e implementa los métodos *addTuioCursor(tcur)*, *updateTuioCursor(tcur)* y *removeTuioCursor(tcur)*. Básicamente su función es construir un objeto de la clase *CursorInfo* (que simplemente contiene la información del cursor y el tipo de evento, ya sea añadir, actualizar o eliminar) y la añade a la cola de eventos.

La clase encargada de manejar los eventos (explicada a continuación) los leerá y realizará la acción correspondiente.

4.3.6. TUIOHandler

Es una de las clases más importantes. Contiene el código necesario para manejar los elementos del sistema. Tiene dos métodos a destacar: *handle()* y *pick()*.

El primero de ellos se encarga de realizar las acciones necesarias cada vez que llega un evento (de los que genera *MyTuioListener*). Para ello analiza el tipo de evento, tiene en cuenta si es el primer o segundo contacto (ya que hay que cumplir lo especificado en la Sección 3.2 punto 6 referente a los gestos multitáctiles) y en función del tipo de evento y número de dedos realiza la acción correspondiente (seleccionar un elemento mediante *pick()*, mover un elemento, rotarlo y agrandarlo, soltarlo sobre el escritorio o sobre un botón, y hacer click en un botón).

El método *pick()* se encarga de seleccionar el elemento correspondiente del grafo de escena para un determinado punto. Es un método genérico disponible en múltiples ejemplos de OSG, pero se ha modificado para obtener sólo los elementos de interés.

Capítulo 5

Resultados

En este capítulo se presentan los primeros resultados obtenidos tras la construcción del sistema.

5.1. Montaje final

En la Figura 5.1 se puede observar el escritorio finalmente construido, que consiste en una mesa de las disponibles en el laboratorio (de 120cm x 80cm), y la estructura que soporta los distintos elementos (proyector, Kinect, y cámara).

El sistema está compuesto por los siguientes componentes:

- Proyector InFocus IN1503 [40].
- Microsoft Kinect for Xbox 360 [25].
- Cámara digital Canon EOS 1100D [34].
- Ordenador de sobremesa. Con las siguientes características técnicas: procesador Intel i5 3570K, 16GB de Ram, tarjeta gráfica ATI HD7750 1GB. Ejecutando el sistema operativo Ubuntu 12.04.

La Figura 5.2 muestra aplicación en funcionamiento sobre la mesa, en ella se observan dos documentos, que tras ser digitalizados utilizando la cámara, han pasado por la fase de "preproceso". Se han recortado y normalizado automáticamente y se pueden leer con bastante claridad una vez ampliados. También se observan los botones que permiten la interacción con la aplicación, y un documento en la esquina pendiente de ser digitalizado y que no interfiere con la aplicación en ejecución.

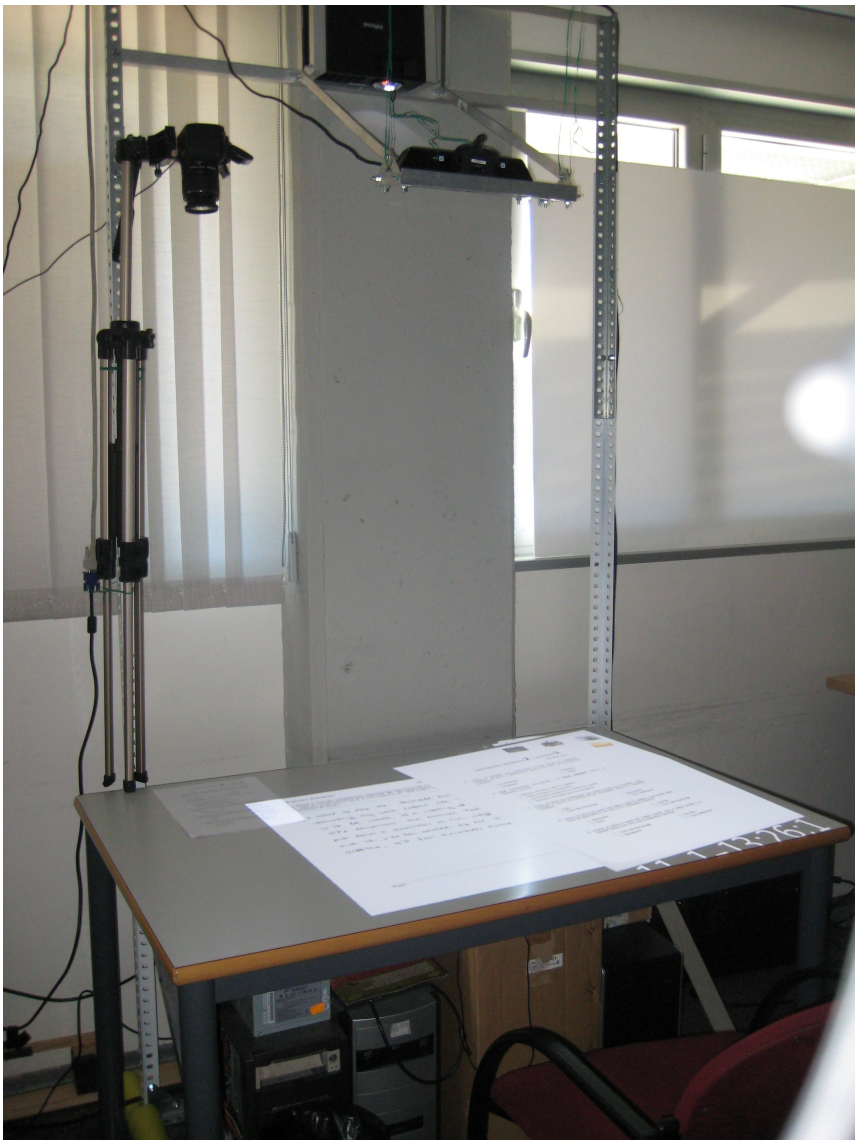


FIGURA 5.1: Montaje final, del prototipo construido en el laboratorio. En la parte superior izquierda de la imagen se observa la cámara de alta resolución para la captura de documentos, en el centro el proyector y Kinect. El sistema ejecutándose muestra dos documentos que se han digitalizado, y un documento aún sin digitalizar en la zona de captura.

La Figura 5.3 muestra la tableta sobre el escritorio. Se puede observar el mismo documento sobre la mesa, digitalizado, y la tableta.

Se puede encontrar un *vídeo (ver archivo adjunto)* del funcionamiento del escritorio.

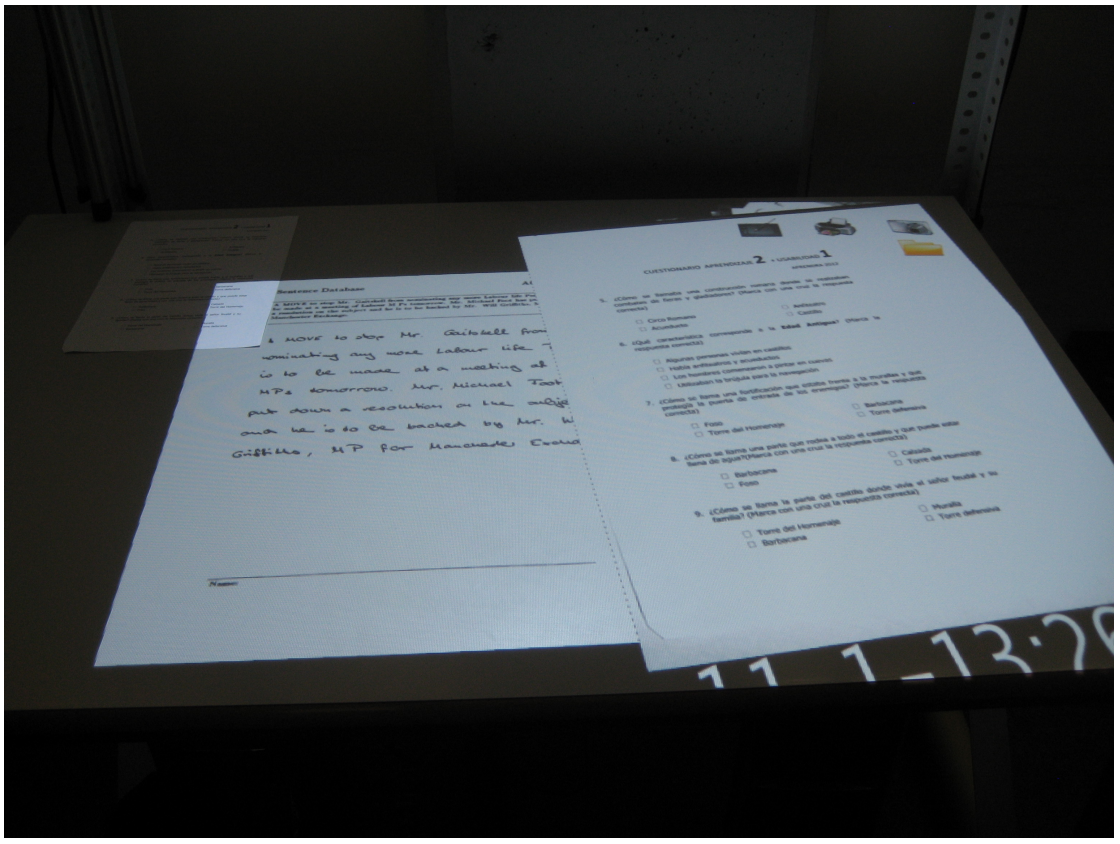


FIGURA 5.2: Aplicación ejecutándose sobre la mesa. En la parte superior un documento pendiente de digitalizar, y en el centro dos documentos abiertos.

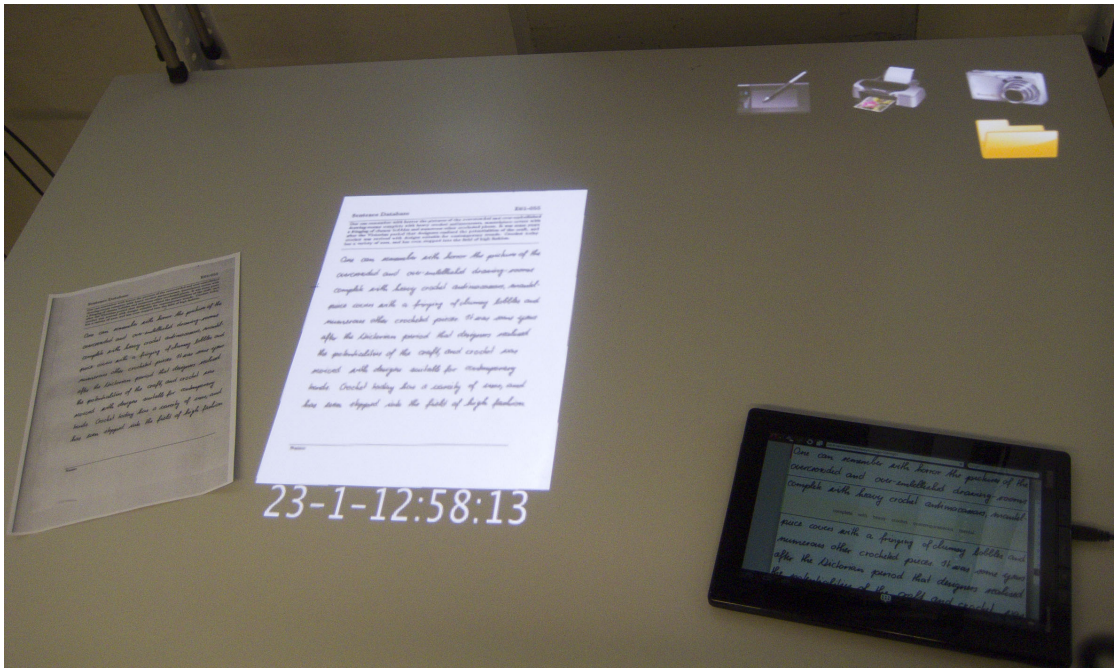


FIGURA 5.3: Un documento en papel, digitalizado y mostrado en el escritorio, y un fragmento del mismo en la tableta.

5.2. Pruebas realizadas

Para evaluar la usabilidad del sistema, se pidió a algunos usuarios que realizasen ciertas tareas. Se les indicó el funcionamiento del reconocedor de dedos, pues tiene ciertas peculiaridades (no reconoce el contacto con la mesa, sino que detecta una mano con un dedo extendido como puntero).

Se pidió a los usuarios que intentasen:

- Capturar dos documentos. Se puede observar en la Figura 5.4.
- Ampliarlos hasta que fuesen visibles.
- Archivarlos en la carpeta, y extraerlos. Se observa en la Figura 5.5.

Las funciones de enviar a la tableta e imprimir no se probaron, pues la interacción con ellos es análoga a archivar un documento en la carpeta y aún no estaban implementadas en el momento de la prueba.

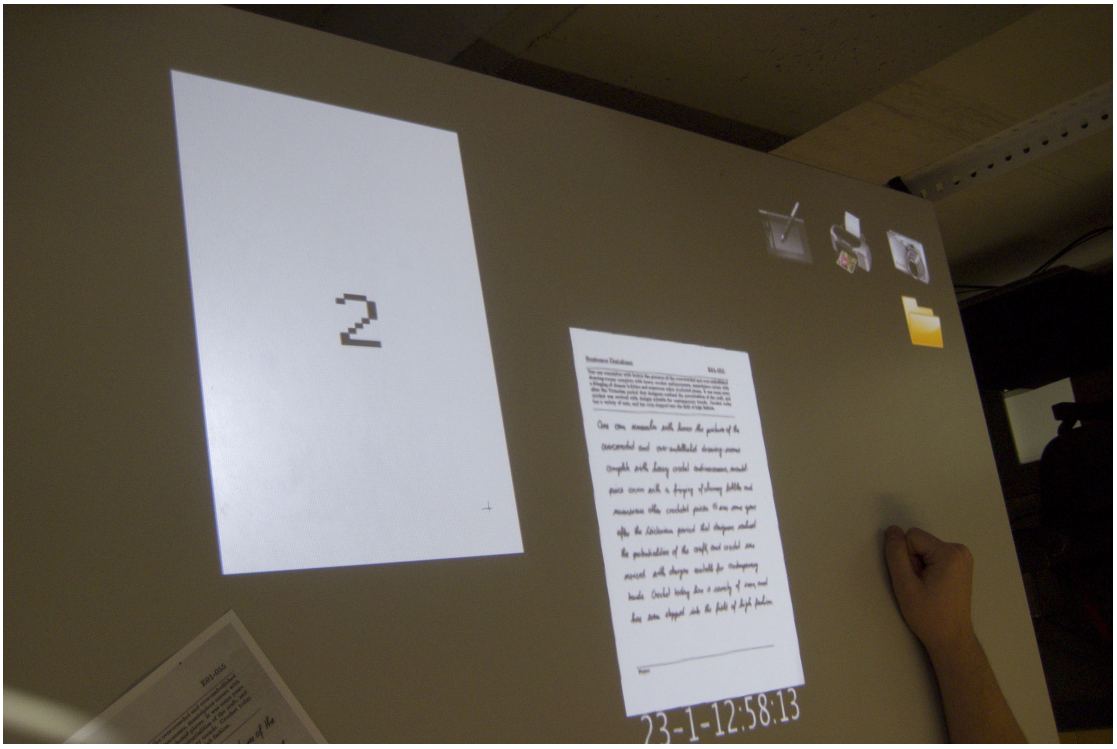


FIGURA 5.4: El escritorio con un documento capturado, y otro en proceso. La cuenta atrás indica el área de captura.

Los usuarios señalaron el buen funcionamiento del sistema a la hora de manipular los documentos, los gestos multitáctiles simplifican bastante su funcionamiento y son una

FIGURA 5.5: Contenido de una carpeta en la que el usuario acaba de archivar dos documentos.

forma natural de interacción. Pero también señalaron una considerable dificultad a la hora de utilizar los botones.

Este problema en parte es debido a lo que se explica en el Apéndice A, concretamente en la Figura A.1b, por el propio funcionamiento del detector de dedos, pues es complicado realizar un *click* con precisión sobre un botón.

Otro problema que señalaron fue que no quedaba claro dónde hay que hacer *click* para extraer el documento de una carpeta. Esto es debido a que se puede seleccionar tanto el icono como el nombre del fichero, pero en el caso del texto sólo se selecciona si el toque coincide sobre el ancho de las letras (no los huecos), lo cual confunde a los usuarios.

Al arrastrar los documentos sobre los botones (o carpetas), algunos usuarios los soltaban en cuanto el documento y el botón se solapaban, en vez de cuando el dedo tocaba el botón (que es la forma en que normalmente se comportan estas acciones en la gran mayoría de sistemas). Seguramente sea causado al no indicar de ninguna manera al usuario si está o no seleccionando el botón. Este problema se resolvería implementando algún tipo de realimentación que indicase al usuario que está seleccionando o no un determinado botón.

La solución a el problema de hacer *click* en los botones podría consistir en cambiar la forma en que se seleccionan, y en vez de *apuntar* sobre ellos, hacer que se activasen tras señalarlos durante cierto tiempo.

El problema que existía al extraer un documento de una carpeta se solucionó haciendo que también se pueda seleccionar los documentos aunque se toque entre los huecos de las letras. De esta forma, toda la fila desde el icono hasta el final del nombre del documento permite extraerlo.

Capítulo 6

Conclusiones

6.1. Conclusiones del trabajo realizado

En este proyecto se ha presentado el prototipo inicial de un sistema que permite la interacción con documentos tanto en formato digital como en formato físico.

El presente prototipo permite la manipulación de documentos digitales de forma natural, utilizando para ello técnicas de interacción multitáctil, reconocimiento de dedos sin ningún tipo de periférico acoplado a las manos, y permitiendo realizar transformaciones de rotación, escalado y traslación para facilitar la lectura de los documentos.

Permite una sencilla conversión de documentos de formato físico a formato digital, utilizando para ello una cámara digital, que es capaz de digitalizar un documento rápidamente. Este se procesa de forma automática para obtener una imagen recortada al contenido del documento y normalizada, para así obtener la mayor calidad posible.

El sistema se apoya en una tableta para editar y corregir con precisión y comodidad los documentos que se han procesado.

El documento digitalizado puede volver a imprimirse con un sencillo gesto, y obtener la versión de mayor calidad posible (el texto reconocido si se ha completado el procesado, o la imagen preprocesada en el caso contrario).

Dado que se trata de una versión inicial del sistema, aún quedan muchos puntos posibles de mejora que se enumeran en el apartado de trabajo futuro.

6.2. Trabajo futuro

De cara a un futuro inmediato, debería resolver los problemas que los usuarios destacaron respecto a la interfaz mencionados al final de la Sección 5.2. También podría pensarse en ampliar la funcionalidad de la aplicación que maneja el Escritorio Virtual, añadiendo funciones, subsanando las limitaciones y mejorando tanto el comportamiento como la apariencia de la aplicación.

Visto los problemas que presenta la detección de dedos, la poca precisión y fiabilidad del sistema, sería interesante volver a analizar y probar otras alternativas para solventar los problemas actuales.

También podría pensarse en cambiar la aplicación actual por una de las mencionadas en la Sección 4.1.1 (ya sea utilizando un entorno gráfico que permita la libertad de movimientos presentados en la aplicación, o añadiendo la opción de cargar ventanas del sistema operativo utilizando VNC). Recientemente Microsoft ha lanzado la nueva versión de su sistema operativo Windows 8, con soporte oficial para superficies táctiles, se podría plantear la posibilidad de desarrollar una nueva revisión del Escritorio Virtual bajo este sistema operativo. Dado que Windows 8 está pensado para usarse mediante una superficie táctil, debería simplificar el diseño de una interfaz de este tipo y proveer de herramientas para realizar aplicaciones con este tipo de interacción.

Otra opción interesante sería analizar otros posibles usos del sistema que se ha construido, estudiando otras tareas que requieran de una gran superficie de trabajo y no mucha resolución.

6.3. Descripción del desarrollo del proyecto

Este proyecto se ha realizado como tesina del Máster en Inteligencia Artificial, Reconocimiento de Formas e Imagen Digital. Su desarrollo comenzó a principios del año 2012, a continuación se mencionan algunos de los problemas que surgieron durante el desarrollo y las posibles soluciones estudiadas (muchas de ellas han acabado descartadas por el camino como se explica en la Sección 4.1.1).

En referencia a la **duración**, tiempo y dedicación invertidos en el proyecto:

- Los primeros meses me dediqué a leer artículos relacionados, documentarme en general, y hacer pequeñas pruebas con las distintas posibilidades encontradas (gran parte de ellas explicadas en la Sección 4.1.1), junto a acabar algunas asignaturas del máster, por lo que la dedicación no fue completa.

- A partir de Mayo de 2012, tras reunirnos todos los miembros del grupo, se definió con exactitud los requisitos y procedí a iniciar el desarrollo del sistema.
- Durante los meses de verano, y hasta el final del proyecto, la dedicación fue a tiempo completo. En este tiempo se han ido realizando diversas reuniones con el resto del equipo para comprobar el estado del desarrollo del proyecto, definir con mayor precisión la forma de comunicación entre los distintos componentes y realizar diversas pruebas de integración del sistema.

Como **experiencia personal**, destacaría que considero haber aprendido mucho respecto al funcionamiento de un grafo de escena como es *OSG*, las diversas técnicas existentes para capturar gestos multitáctiles, la programación de la interacción con sistemas no muy comunes como es el presentado, y la integración de distintas áreas. También considero interesantes este tipo de proyectos multidisciplinares, ya que permiten relacionar equipos de investigación de distintas áreas, aunque también conllevan ciertos problemas añadidos. La integración de las distintas partes que componen el proyecto en muchos casos no es trivial, más cuando los distintos componentes del proyecto son a su vez otros proyectos que, o bien ya estaban realizados de una forma que dificulta su integración, o bien estaban aún en desarrollo.

Como aspectos negativos destacaría que pese a considerarme un fiel defensor del "Software Libre", en este proyecto he acabado bastante "quemado" y aseguraría que he perdido mucho tiempo en gran parte a causa de los problemas con Linux, y con OSG, que menciono a continuación.

Respecto a los **problemas** encontrados y sus soluciones, me gustaría destacar los siguientes:

- **Problemas con Linux**, durante el desarrollo del proyecto he encontrado muchos tipos de problemas, relacionados en gran parte con el sistema operativo:
  - Problemas con *drivers*, ya que para el proyecto se requiere cierta aceleración gráfica, y los drivers "open-source" incluidos con la gran mayoría de distribuciones de Linux no tienen el rendimiento esperado, procedí a instalar los drivers "propietarios" proporcionados por el fabricante de la tarjeta gráfica (*Nvidia* en el caso del ordenador en el que realicé el desarrollo). Al instalarlos la librería "TUIO" dejó de funcionar (y se solucionó desinstalándolos y volviendo a los drivers open-source). Sospecho que tiene algo que ver con los problemas documentados que existen entre los drivers y la librería *pthreads* `http://us.download.nvidia.com/XFree86/Linux-x86/173.14.09/README/chapter-09.html`

  - Problemas con dependencias, que hacen que instalar ciertos programas sea un martirio, más si dicho programa aún no está en fase "stable" y hay que instalar algunas librerías "beta", no es raro acabar con un sistema altamente inestable. La solución más rápida pasa por eliminar y volver a instalar el SO (cambiar una librería *beta* por una versión suya anterior estable no es algo trivial). Tuve que realizarlo varias veces durante la fase de pruebas.
  - Problemas con actualizaciones, al igual que antes, una actualización del sistema puede acarrear actualización de librerías. Algunos programas llegaron a dejar de compilar tras actualizar. La solución acaba pasando por no actualizar bajo ninguna circunstancia.

- **Problemas con OSG**, de muy diverso tipo. Desde problemas al compilar producidos durante la última fase del proceso (por lo que terminaba con algunas horas perdidas y una frustración considerable, sobretodo cuando el fallo es por una de las mencionadas actualizaciones de librerías), hasta partes del propio *OSG* que están sin implementar o en una fase muy inicial de su desarrollo, como por ejemplo *OSGWidgets* (descrito en [41] pág. 378 y que hubiese permitido implementar fácilmente los botones del sistema), sin olvidar partes que están implementadas, sobre las que apenas hay documentación. La solución al primer caso pasa por instalar OSG desde los repositorios (siempre y cuando no se necesite modificar nada ni se requiera alguna función un poco especial, como sucedía en la primera fase de pruebas), y en el segundo caso documentándose bien antes de utilizar cualquier función, y respecto a las partes sin documentar normalmente descargando el código fuente puede intuirse su funcionamiento.

- Uno de los "problemas" de OSG que más me ha costado solucionar reside a la hora de añadir o modificar dinámicamente nodos en tiempo de ejecución. Hay que tener mucho cuidado al realizar estas operaciones, ya que debido a el procesamiento multihilo, una operación que podría parecer sencilla acarrea muchos problemas si no se realiza en el momento adecuado (durante la fase de "update" del nodo que queremos modificar), y el resultado es una aplicación que puede dar fallos de segmentación de forma aleatoria. La solución pasa por utilizar *Callbacks* para actualizar en la fase correspondiente, la técnica está descrita en el capítulo 8 del libro [41], y hay que tener en cuenta que sólo podemos modificar el nodo que realiza el *callback* ya que si modificamos otro nodo distinto durante este proceso, también estamos incurriendo en el mismo fallo.

- **Problemas con Kinect**, principalmente debidos a el ruido y artefactos producidos en forma de bandas verticales, reconocidos en el propio *FAQ* del *SDK* oficial [29]. La solución a estos problemas consistió en ampliar el margen de error admitido por

el detector de dedos, aunque de esta forma se empeoró el rendimiento y se detecta el dedo mucho antes de tocar la superficie.

- **Problemas con la detección de dedos**, ya que como he comentado anteriormente, debido en parte a las limitaciones de Kinect, su precisión es reducida. La solución consistió en aumentar los iconos de la interfaz de usuario hasta que fuesen fácilmente seleccionables y cambiar algunos gestos para que en vez de necesitar hacer "click" (*tocar* y *soltar* dentro del icono) se detectase la acción tanto al *tocar* como al *soltar*, de esta forma los botones responden mejor. También hubo que aplicar un filtro debido a que la señal de las posiciones de los dedos era muy inestable, está explicado en el Apéndice A.

- **Problemas para encontrar una cámara compatible**, aunque se nos prestó una, era interesante conseguir una exclusivamente para el proyecto. El problema principal consiste en que las cámaras de bajo coste no permiten tomar fotos desde un ordenador, la cámara prestada tenía un firmware no oficial que permitía hacerlo, pero en los modelos disponibles a la venta actualmente ninguno era compatible. La solución consistió en encontrar una cámara económica que fuese compatible con la captura remota del software *gPhoto* [33], de las disponibles a la venta elegí la Canon 1100D por ser la más económica, tras contactar con el soporte técnico de Canon para confirmar si la cámara podía hacer lo que se necesitaba recibiendo por respuesta una negativa, acabé preguntando en un foro de usuarios de dicho modelo donde confirmaron que funciona.

Abreviaturas y Definiciones

A continuación se definen brevemente algunas palabras, abreviaturas o conceptos que se utilizan a lo largo de la memoria. Algunos están detallados más detenidamente en apartados posteriores, pero se describen para facilitar la comprensión de la memoria.

OSG **O**pen **S**cene **G**raph es un grafo de escena 3D, de código abierto y con un alto rendimiento. Está escrito completamente en C++ y OpenGL y funciona sobre la gran mayoría de sistemas operativos. Puede utilizarse para proyectos de simulación, juegos, realidad virtual, visualización científica, etc. [31]

SDK **S**oftware **D**evelopment **K**it, se refiere al conjunto de herramientas que permiten crear aplicaciones para un sistema concreto.

Escritorio Virtual de entre las muchas acepciones, en este proyecto se entiende como un espacio de trabajo en el que se muestran los documentos, objetos, etc. virtuales con los que trabajaremos, renderizados por un ordenador, simulando el comportamiento de estos elementos en el mundo real.

Renderizar generar una imagen por ordenador desde un modelo.

Multitáctil se refiere a la tecnología (tanto software, como hardware que lo permite) capaz de reconocer e interpretar simultáneamente múltiples puntos de contacto.

Segmentación de texto se refiere al proceso de reconocimiento de texto (ya sea realizado por un humano, o por un ordenador como en el caso de esta aplicación) mediante el cuál un texto se analiza y descompone en unidades con significado (letras, palabras, oraciones, etc).

Realidad Aumentada técnica que mediante el uso de simulación por computador permite introducir objetos virtuales en sobre una información física ya existente. Para

ello utiliza dispositivos como pantallas y cámaras, gafas, etc. que permiten mostrar una escena física real sobre la que se superponen objetos virtuales.

Tracker se refiere a cualquier tipo de herramienta que permita seguir un determinado elemento del mundo real, ya sea un marcador como los utilizados en realidad aumentada, un elemento físico que se ponga sobre el cuerpo de la persona como los utilizados en realidad virutal o algún sistema mediante cámaras que permita reconocer y seguir el movimiento de un determinado elemento.

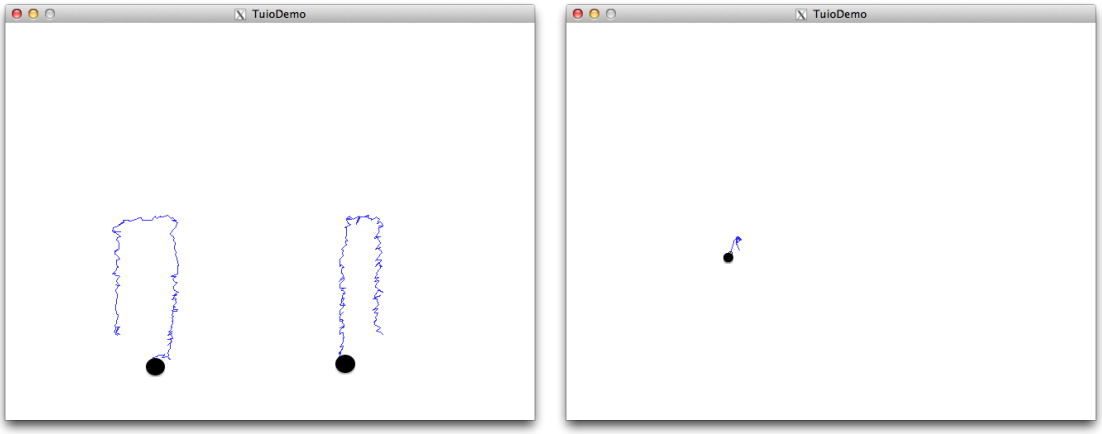
Estereoescopía técnica mediante la cual se consigue una imagen tridimiensional capaz de crear una sensación de profundidad.

Tablet o tableta, es un tipo de ordenador potátil de pequeño tamaño integrado en una pantalla, en el que se interactúa principalmente sobre su pantalla con los dedos o un lápiz.

Apéndice A

Filtrado de datos del reconocedor de dedos

En este apéndice se explica brevemente los problemas que presenta la señal obtenida del módulo de reconocimiento de dedos, y cómo se ha estabilizado el movimiento utilizando un filtro de Kalman [42].



(A) Dos dedos (B) "Click"

FIGURA A.1: Reconocimiento de dedos, señal sin filtrar. La Figura A.1a muestra la trayectoria seguida por el cursor a lo largo del tiempo al realizar un movimiento en forma de " ⊓ " con cada dedo, finalizando en la parte inferior central (donde se ve un círculo con la última posición detectada). La Figura A.1b muestra el movimiento del cursor al realizar un "click".

La Figura A.1a muestra el recorrido seguido por el cursor a lo largo del tiempo al realizar un movimiento sencillo en forma de " ⊓ " con dos dedos. Se observa su irregularidad a

lo largo del recorrido, que comienza en la parte exterior y finaliza en el círculo que indica la última posición del dedo.

En la Figura A.1b se puede observar como al realizar un "click" el cursor describe un movimiento en forma de pico. Comienza en la parte inferior derecha, se mantiene un tiempo en el punto donde se estaba señalando, y finaliza en la parte inferior izquierda (con el círculo que indica la última posición detectada).

Esto provoca una gran dificultad de interacción con el sistema presentado en esta tesina, ya que si se utiliza esta señal directamente, los objetos "temblarían" al seleccionarlos y manipularlos, y sería muy complicado e incómodo interaccionar con el sistema. Se decidió por ello implementar un filtro que suavizase el movimiento.

Para la implementación del filtro de Kalman se utilizó la librería OpenCV [43] que provee de una clase y métodos [44] que permiten implementar de forma fácil y rápida un filtro de Kalman. Este filtro está implementado en la clase *MyTuioListener* que es la encargada de crear el cliente TUIO y enviar los puntos al grafo de escena, de forma que se puede implementar ahí el filtro y que afecte directamente a todos los datos que recibe la aplicación.

De esta forma, aunque se produce un ligero retraso al realizar los movimientos, el movimiento que se consigue es mucho más suave, como se puede observar en el *vídeo (ver archivo adjunto)*.

Apéndice B

Manual de usuario, instrucciones de compilación, y consideraciones respecto a la integración con otros componentes

En este apéndice se presenta un breve manual de uso de la aplicación, a continuación las instrucciones a seguir para compilar la aplicación, y por último unas breves consideraciones sobre la integración con los componentes del resto del equipo.

B.1. Manual de usuario

Para ejecutar la aplicación debemos abrir un terminal en la carpeta que contiene el ejecutable *EscritorioVirtual* y las carpetas *resources y Documents* donde se almacenan ciertos ficheros del programa, y los documentos escaneados respectivamente.

En dicha carpeta ejecutamos: `./EscritorioVirtual` y se lanzará la aplicación, por defecto a pantalla completa. Podemos definir antes de ejecutarlo la variable de entorno `OSG_WINDOW = x y width height` siendo x e y la posición superior izquierda de la ventana, *width* y *height* el tamaño en ancho y alto de la ventana. También se puede definir

que se utilice *antialiasing* para suavizar los bordes de las imágenes utilizando el argumento `-multiSamples n`, aunque dependiendo de la tarjeta gráfica o sus controladores puede causar que el programa no arranque.

Se puede probar el funcionamiento de la aplicación utilizando en vez del reconocedor de dedos, el programa *SimpleSimulator*, disponible en la librería *TUIO*, descargando y compilando el fichero *TUIO_ CPP.zip* disponible en [45].

Una vez arrancada la aplicación obtenemos una ventana como la que aparece en la Figura B.1, donde se puede observar el escritorio con dos documentos digitalizados y los botones. Podemos interaccionar con ella con los gestos descritos en Sección 4.1.4.

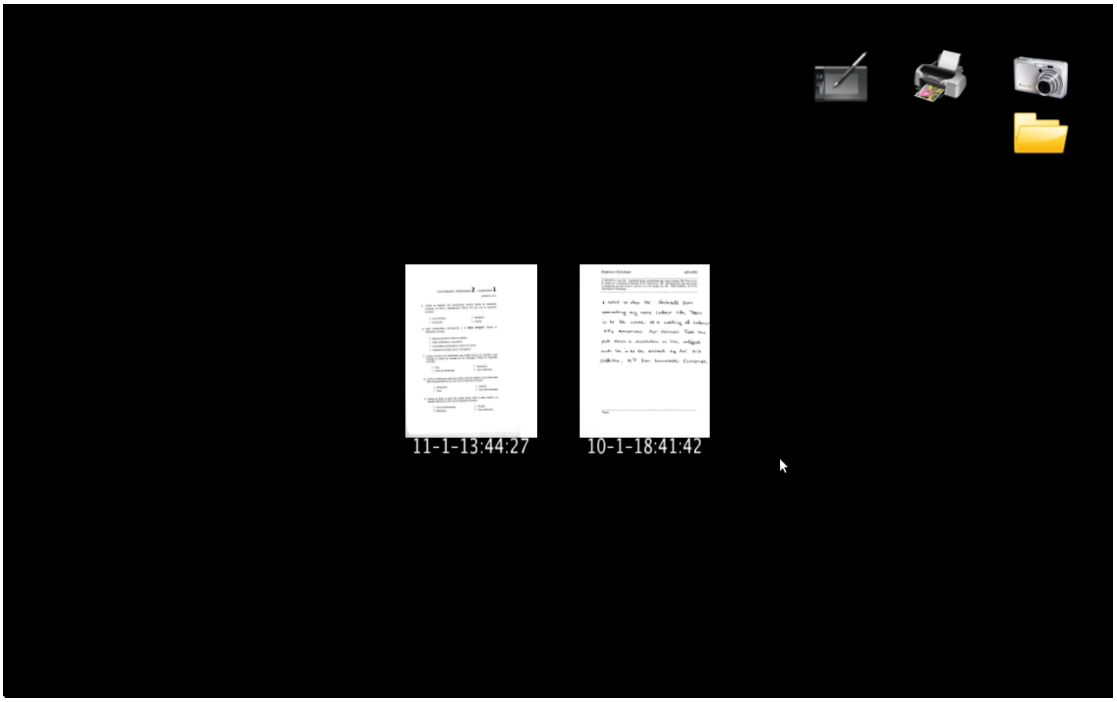


FIGURA B.1: Captura de pantalla de la aplicación, se muestran dos documentos digitalizados en el centro, y los botones en la parte superior derecha.

Si arrastramos uno de ellos sobre la carpeta, y seguidamente tocamos la carpeta para ver su contenido el resultado sería el mostrado en la Figura B.2.

Si tocamos el documento que se muestra en la carpeta este aparecerá de nuevo en el escritorio, y si tocamos con un dedo y seguidamente tocamos con otro dedo en la mesa, el documento podrá ampliarse y rotarse, tal y como muestra la Figura B.3.

La interacción con el resto de botones es similar. El de la cámara tomará una foto si se toca, y el de la tablet e impresora al dejar un documento sobre ellos.

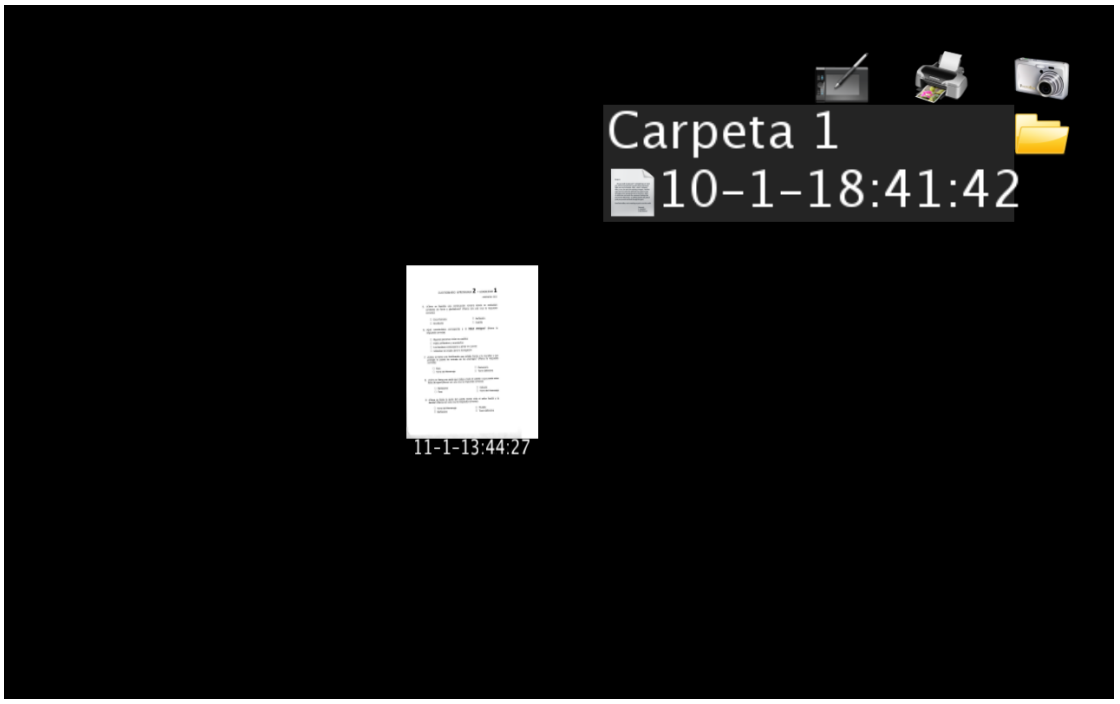


FIGURA B.2: Captura de pantalla de la aplicación, se muestra un documento en el centro, y el contenido de una carpeta.

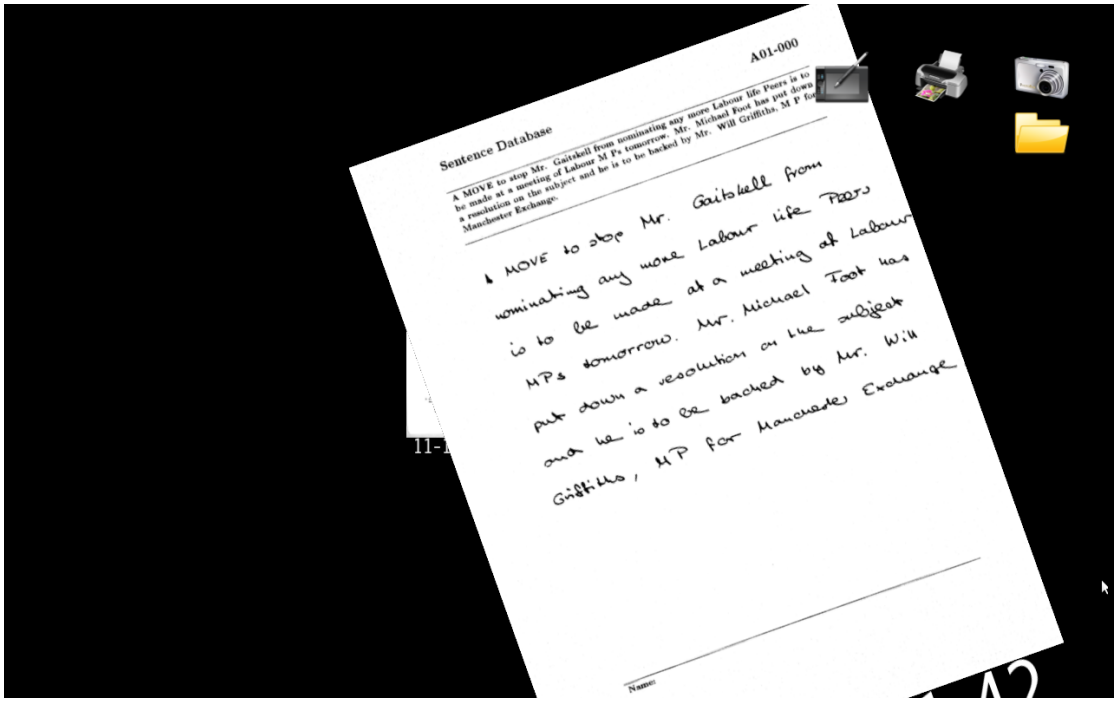


FIGURA B.3: Captura de pantalla de la aplicación, se muestra un documento que está siendo manipulado, ampliándolo para leerlo con mayor facilidad.

B.2. Instrucciones de compilación

Para compilar y ejecutar el proyecto, partiendo de un sistema operativo Linux, necesitamos instalar las siguientes librerías y programas:

- *CMake*
- *OSG*
- *OpenCV*
- *gPhoto2*
- *SDL*
- *ImageMagick*

En un sistema con un sistema de repositorios estilo *Debian* podemos instalarlas de la siguiente forma desde un terminal:

```
# sudo apt-get install cmake libopenscenegraph-dev libopencv-dev
gphoto2 libsdl1.2-dev imagemagick
```

Para compilar el proyecto vamos a la carpeta *build*, y ejecutamos en un terminal:

```
# cmake .. && make
```

Compilará el proyecto, y una vez finalizado podremos ejecutarlo con el comando:

```
# ./EscritorioVirtual
```

B.3. Consideraciones respecto a la integración

En esta sección se explica cómo se ha realizado la integración con los componentes de otros miembros del equipo, principalmente con el módulo de procesado de texto.

Este procesado se realiza en dos partes, separados en dos *scripts* diferentes: `preproceso.sh` y `HTR.sh` (*handwritten text recognition*). Al realizar la captura de la imagen del documentos se llama al primero de ellos, y cuando este finaliza se añade el Documento con la

imagen ya recortada y normalizada al grafo de escena y se lanza el segundo *script* (que procesará el texto para reconocerlo).

También indicar que para la implementación de los botones que faltan, hay que añadir el código necesario dentro la función `dropAction` o `clickAction` correspondiente a cada uno de ellos. Hay que tener en cuenta que si la acción a realizar es lanzar un determinado *script*, programa o una acción que requiera de cierto tiempo de procesado, habría que lanzarla o bien como un proceso aparte (como se realiza a la hora de tomar una captura y cuyo código se puede observar en *CameraController.cpp* y *CaptureButton.cpp*) o bien creando un nuevo hilo de ejecución, ya que sino bloquearía la ejecución del programa.

Bibliografía

[1] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. User-defined gestures for surface computing. In *Proceedings of the 27th international conference on Human factors in computing systems*, CHI '09, pages 1083–1092, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-246-7. doi: 10.1145/1518701.1518866. URL http://doi.acm.org/10.1145/1518701.1518866.

[2] Kenrick Kin, Tom Miller, Björn Bollensdorff, Tony DeRose, Björn Hartmann, and Maneesh Agrawala. Eden: a professional multitouch tool for constructing virtual organic environments. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 1343–1352, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979141. URL http://doi.acm.org/10.1145/1978942.1979141.

[3] Martin Hachet, Benoit Bossavit, Aurélie Cohé, and Jean-Baptiste de la Rivière. Toucheo: multitouch and stereo combined in a seamless workspace. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pages 587–592, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0716-1. doi: 10.1145/2047196.2047273. URL http://doi.acm.org/10.1145/2047196.2047273.

[4] Microsoft® PixelSense™ Website - How Works, 2012. URL http://www.microsoft.com/en-us/pixelsense/pixelsense.aspx.

[5] Samsung SUR40 website, 2012. URL http://www.samsung.com/mx/consumer/monitor-peripherals-printer/visual-solutions/touch-screens/LH40SFWTGC/ZA.

[6] nSquared Presenter 2.0, video @Engadget., 2012. URL http://www.engadget.com/2012/03/01/nsquareds-seamless-computing-ties-windows-surface-and-ipads-v/.

[7] Andrew D. Wilson. Playanywhere: a compact interactive tabletop projection-vision system. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 83–92, New York, NY, USA, 2005. ACM. ISBN 1-59593-271-2. doi: 10.1145/1095034.1095047. URL `http://doi.acm.org/10.1145/1095034.1095047`.

[8] M. Kobayashi and H. Koike. Enhanceddesk: integrating paper documents and digital documents. In *Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific*, pages 57 –62, jul 1998. doi: 10.1109/APCHI.1998.704149.

[9] Anand Agarawala and Ravin Balakrishnan. Keepin' it real: pushing the desktop metaphor with physics, piles and the pen. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, CHI '06, pages 1283–1292, New York, NY, USA, 2006. ACM. ISBN 1-59593-372-7. doi: 10.1145/1124772.1124965. URL `http://doi.acm.org/10.1145/1124772.1124965`.

[10] Malte Weiss, Simon Voelker, Christine Sutter, and Jan Borchers. Benddesk: dragging across the curve. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 1–10, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0399-6. doi: 10.1145/1936652.1936654. URL `http://doi.acm.org/10.1145/1936652.1936654`.

[11] Eduardo Martorell, Armando de la Re, and Francisco Abad. Diseño y construcción de un banco de trabajo multitáctil de bajo coste. In *Conference: Congreso Español de Informática Gráfica*, September 2010. URL `https://jira.ai2.upv.es/confluence/pages/viewpage.action?pageId=11239438`.

[12] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST '05, pages 115–118, New York, NY, USA, 2005. ACM. ISBN 1-59593-271-2. doi: 10.1145/1095034.1095054. URL `http://doi.acm.org/10.1145/1095034.1095054`.

[13] Meredith Ringel Morris, Jarrod Lombardo, and Daniel Wigdor. Wesearch: supporting collaborative search and sensemaking on a tabletop display. In *Proceedings of the 2010 ACM conference on Computer supported cooperative work*, CSCW '10,

pages 401–410, New York, NY, USA, 2010. ACM. ISBN 978-1-60558-795-0. doi: 10.1145/1718918.1718987. URL http://doi.acm.org/10.1145/1718918.1718987.

[14] Hrvoje Benko, Edward W. Ishak, and Steven Feiner. Cross-dimensional gestural interaction techniques for hybrid immersive environments. In *Proceedings of the IEEE Virtual Reality. Conference*, pages 209–216, 2005.

[15] Hrvoje Benko, Ricardo Jota, and Andrew Wilson. Miragetable: freehand interaction on a projected augmented reality tabletop. In *Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems*, CHI '12, pages 199–208, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1015-4. doi: 10.1145/2207676.2207704. URL http://doi.acm.org/10.1145/2207676.2207704.

[16] Ricardo Jota and Hrvoje Benko. Constructing virtual 3d models with physical building blocks. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '11, pages 2173–2178, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0268-5. doi: 10.1145/1979742.1979915. URL http://doi.acm.org/10.1145/1979742.1979915.

[17] MirageBlocks - project Website, 2012. URL http://research.microsoft.com/en-us/projects/mirageblocks/.

[18] S. Di Verdi, D. Nurmi, and T. Hollerer. Arwin - a desktop augmented reality window manager. In *Mixed and Augmented Reality, 2003. Proceedings. The Second IEEE and ACM International Symposium on*, pages 298 – 299, oct. 2003. doi: 10.1109/ISMAR.2003.1240729.

[19] Aurélie Cohé, Fabrice Dècle, and Martin Hachet. tbox: a 3d transformation widget designed for touch-screens. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, CHI '11, pages 3005–3008, New York, NY, USA, 2011. ACM. ISBN 978-1-4503-0228-9. doi: 10.1145/1978942.1979387. URL http://doi.acm.org/10.1145/1978942.1979387.

[20] Mark Hancock, Thomas ten Cate, and Sheelagh Carpendale. Sticky tools: full 6dof force-based interaction for multi-touch tables. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 133–140, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-733-2. doi: 10.1145/1731903.1731930. URL http://doi.acm.org/10.1145/1731903.1731930.

[21] Jason L. Reisman, Philip L. Davidson, and Jefferson Y. Han. A screen-space formulation for 2d and 3d direct manipulation. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST '09, pages 69–78, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-745-5. doi: 10.1145/1622176.1622190. URL http://doi.acm.org/10.1145/1622176.1622190.

[22] Paul Dietz and Darren Leigh. Diamondtouch: a multi-user touch technology. In *Proceedings of the 14th annual ACM symposium on User interface software and technology*, UIST '01, pages 219–226, New York, NY, USA, 2001. ACM. ISBN 1-58113-438-X. doi: 10.1145/502348.502389. URL http://doi.acm.org/10.1145/502348.502389.

[23] Microsoft® Surface Hardware components, 2012. URL http://technet.microsoft.com/en-us/library/ee692114(v=surface.10).aspx#bkmk_HardwareComponents.

[24] A. Agarwal, S. Izadi, M. Chandraker, and A. Blake. High precision multi-touch sensing on surfaces using overhead cameras. In *Horizontal Interactive Human-Computer Systems, 2007. TABLETOP '07. Second Annual IEEE International Workshop on*, pages 197 –200, oct. 2007. doi: 10.1109/TABLETOP.2007.29.

[25] Kinect official website, 2012. URL http://www.microsoft.com/en-us/kinectforwindows/.

[26] Andrew D. Wilson. Using a depth camera as a touch sensor. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 69–72, New York, NY, USA, 2010. ACM. ISBN 978-1-4503-0399-6. doi: 10.1145/1936652.1936665. URL http://doi.acm.org/10.1145/1936652.1936665.

[27] Kinect for Windows Sensor Components and Specifications, 2012. URL http://msdn.microsoft.com/en-us/library/jj131033.aspx.

[28] P. Lisouski A.K. Mortensen M.K. Hansen T. Gregersen M.R. Andersen, T. Jensen and P. Ahrendt. In *Technical Report ECE-TR-6*. Aarhus University©, February 2012. URL http://eng.au.dk/fileadmin/DJF/ENG/PDF-filer/Tekniske_rapporter/Technical_Report_ECE-TR-6-samlet.pdf.

[29] Kinect FAQ, Depth artifacts, recommended hardware, ..., 2012. URL http://msdn.microsoft.com/en-us/library/jj131032.aspx#ID4ELC.

[30] Wayland website, 2012. URL http://wayland.freedesktop.org.

[31] OpenSceneGraph Website, 2012. URL http://www.openscenegraph.org.

[32] CHDK - Canon Hack Development Kit webpage, 2013. URL http://chdk.wikia.com/wiki/CHDK.

[33] gPhoto Remote controlling cameras, 2012. URL http://www.gphoto.org/doc/remote/.

[34] Canon EOS 1100D - Webpage, 2012. URL http://www.canon.es/For_Home/Product_Finder/Cameras/Digital_SLR/EOS_1100D/.

[35] Objetivo Canon EF-S 18-55mm f/3.5-5.6 IS II - características, 2013. URL http://www.canon.es/For%5FHome/Product%5FFinder/Cameras/EF%5FLenses/EF%2DS/EF%2DS%5F18%2D55mm%5Ff3.5%2D5.6%5FIS%5FII/.

[36] Kinect field of view @ OpenKinect webpage, 2012. URL http://openkinect.org/wiki/Imaging_Information#Depth_camera_accuracy.

[37] Projector Central - find projector by feature, 2012. URL http://www.projectorcentral.com/projectors.cfm.

[38] M. Kaltenbrunner, T. Bovermann, R. Bencina, and E. Costanza. Tuio a protocol for table-top tangible user interfaces. 2005. URL http://mtg.upf.edu/files/publications/07a830-GW2005-KaltenBoverBencinaConstanza.pdf.

[39] Martin Kaltenbrunner. reactivision and tuio: a tangible tabletop toolkit. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, ITS '09, pages 9–16, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-733-2. doi: 10.1145/1731903.1731906. URL http://doi.acm.org/10.1145/1731903.1731906.

[40] InFocus IN1503 characteristics, 2013. URL http://www.projectorcentral.com/InFocus-IN1503.htm.

[41] Rui Wang and Xuelei Qian. *OpenSceneGraph 3 Cookbook*. Packt Publishing, 2012. ISBN 9781849516884.

[42] Greg Welch and Gary Bishop. An introduction to the kalman filter. Technical report, Chapel Hill, NC, USA, 1995.

[43] OpenCV webpage, 2013. URL `http://opencv.org`.

[44] OpenCV Kalman filter documentation, 2013. URL `http://docs.opencv.org/modules/video/doc/motion_analysis_and_object_tracking.html#kalmanfilter`.

[45] TUIO webpage software download, 2013. URL `http://www.tuio.org/?software`.

[46] Rui Wang and Xuelei Qian. *OpenSceneGraph 3.0: Beginner's Guide*. Packt Publishing, 2010. ISBN 9781849512824.