Interacción con contenidos multimedia mediante Leap Motion

Daniel García Pampín

Resumen— A lo largo de nuestra historia, el ser humano siempre ha buscado la comodidad y facilidad de uso a la hora de trabajar con diferentes sistemas y maquinaria. Desde la aparición de los ratones, permitiéndonos realizar movimientos más precisos, hasta las interfaces gráficas de usuario que facilitan la gestión y presentación de los contenidos. Todas estas tecnologías no solo han mejorado la interacción con los contenidos, sino que, además, han servido de nexo permitiendo a un público más amplio acceder a ellas. Uno de los campos que está ganando más popularidad es el de la interacción con contenidos mediante la captura de movimientos. Este campo abre un sinfín de posibilidades a la hora de interactuar con objetos y nuestro entorno. El objetivo de este artículo es ver las posibilidades que esta tecnología ofrece al trabajar con contenidos multimedia y analizar sus ventajas e inconvenientes.

Palabras clave— Análisis de rendimento, interfaces de computadores – Navegadores, interacción humana con computadores, reconocimiento de gestos, periféricos de ordenadores.

Abstract— Throughout history, humans have been looking for convenience and ease of use when working with different systems and machines. From the mouses appearance, allowing us to perform more accurate movements, to the graphical user interfaces that facilitate the management and presentation of the content. All these tecnologies have not only improved the interaction with content but also they served as a link, allowing a wider audience to access them. One of the fields that is gaining more popularity is the interaction with content through motion capture. This field opens endless possibilities when interacting with objects and our environment. The purpose of this paper is to see the possibilities that this technology brings us when working with multimedia content and analize its advantages and drawbacks.

Index Terms— Computer interfaces - Browsers, computer peripherals, gesture recognition, human computer interaction, performance analysis.

1. Introducción

A lo largo de la historia han surgido diferentes métodos y formas de interactuar con las máquinas, así como periféricos que facilitaban esta interacción. La aparición de interfaces gráficas de usuario, que proliferaron durante los 80 y 90 [1], o los ratones informáticos, son algunos de los elementos que han buscado facilitar esta interacción, ya sea aportando comodidad de trabajo o precisión en los movimientos. La entrada de este tipo de elementos no solo aceleró la velocidad de trabajo, sino que, además, sirvió de nexo permitiendo a un público más amplio con conocimientos informáticos limitados a acceder a estos sistemas.

Durante los últimos años y gracias a los avances computacionales han aparecido nuevas tecnologías como la captura de movimientos y la realidad virtual, las cuales abren un sinfín de posibilidades a la hora de interactuar, ya no solo con las computadoras, sino también con nuestro entorno. Campos como la sanidad, la aviación, el ocio o la educación, son algunos de los que pueden sacar provecho a esta tecnología [2].

Este paper centrará sus esfuerzos en trabajar con Leap Motion, un dispositivo con sensor de movimiento lanzado en 2013 que es capaz de rastrear la posición de manos y dedos y devolver las coordenadas en las que se encuentran.

1.1. Objetivos

El objetivo del proyecto es ver las posibilidades que la tecnología de captura de movimiento ofrece a la hora de interactuar con diferentes tipos de contenidos multimedia (fotos, sonidos, objetos, etc.) en un entorno de navegador, analizar las ventajas e inconvenientes del uso de este tipo de dispositivos e investigar qué nuevas utilidades y funcionalidades pueden aportar en este campo.

1.2. Estructura del paper

La estructura del resto del paper se organiza de la siguiente manera: en la Sección 2 se hará un análisis del estado actual del arte, viendo los diferentes dispositivos de captura de movimiento disponibles en el mercado y algunas de las ventajas y desventajas del dispositivo escogido para la realización de este trabajo. La Sección 3 hablará de la metodología seguida para la realización del trabajo y del porqué se ha escogido. Las Secciones 4, 5, 6 y 7 recorrerán partes vitales en el proceso de desarrollo del software. En ellas, se hablará del proceso de recogida de información, la selección de requerimientos, el diseño del sistema o las pruebas de software realizadas a lo largo del proyecto. A continuación, en las Secciones 8, 9, 10 y 11 se mostrarán las diferentes pruebas realizadas sobre los contenidos y los resultados. Finalmente, la Sección 12 servirá como conclusión del paper. En ella, se resumen los principales puntos tratados y qué conclusiones se han extraído del uso de este tipo de

dispositivos al interactuar con contenidos multimedia.

2. Estado del arte

Antes de profundizar en el proyecto se ha realizado un análisis del estado del arte, teniendo en cuenta dos apartados: contexto actual de mercado y proyectos previos en los que Leap Motion ha sido utilizado.

2.1. Contexto actual

En la actualidad, existen múltiples dispositivos y métodos de captura de movimiento, cada uno de ellos con ventajas e inconvenientes. Es importante, por lo tanto, hacer un análisis de la competencia para ver las diferencias entre los distintos dispositivos.

Si bien existen varios tipos de tecnologías como los *head-mounted display (HMD)* o la captura de movimientos mediante trajes con sensores [3], en este proyecto se ha trabajado con la tecnología de detección de movimientos basada en infrarrojos, concretamente en el rastreo de manos y dedos mediante Leap Motion. Leap Motion es un sensor de movimiento que funciona mediante ondas infrarrojas y que captura los movimientos de manos i dedos que ocurren dentro de su rango de interacción, el cual tiene forma de prisma cuadrangular.

Algunas de las ventajas de Leap Motion son lo poco intrusivo que es el dispositivo a diferencia de otras tecnologias (HMD, trajes con sensores, etc.) o la disposición de una tercera dimensión en contraposición a otros periféricos como el ratón. Sin embargo, Leap Motion tiene claras desventajas en otros aspectos como el medioambiental (condiciones de iluminación, manchas en los sensores, etc.), la necesidad de calibración, limitaciones espaciales u objetos que obstruyen el campo de visión del aparato.

A parte de analizar las ventajas e inconvenientes de las diferentes tecnologías, también se ha realizado un estudio previo de dispositivos de la competencia que trabajan de forma similar a Leap Motion, como Kinect for Xbox de Microsoft o Intel RealSense de Intel. Se han comparado diferentes características como precio, rango de los sensores, lenguajes de programación soportados, etc.

Por último, cabe destacar la importancia y el auge de los contenidos 3D en navegadores. Leap Motion puede aprovechar al máximo esta tercera dimensión facilitada por tecnologías como WebGL [4].

2.2. Otros papers

El último paso antes de empezar el proyecto ha sido investigar en que otros campos y proyectos ha sido útil Leap Motion. Para ello, se han estudiado otros paper disponibles en la web. Esto permite identificar problemas potenciales con los que otras personas se han tenido que enfrentar y posibles soluciones a ellos, así como consideraciones a tener en cuenta cuando se trabaja con el dispositivo.

Algunos ejemplos de papers donde Leap Motion jugaba un rol principal son: "*The Leap Motion controller: a view on sign language*" [5] donde se utiliza el dispositivo para detectar gestos del lenguaje de signos, "*Intuitive and Adaptive Robotic Arm Manipulation using the Leap Motion Controller*" *[6]* donde se controla una mano robótica interactuando con Leap Motion o "*Evaluation of the Leap Motion Controller as a New Contact-Free Pointing Device" [7]* en el cual el autor realiza un análisis de rendimiento del dispositivo.

3. Metodología de trabajo

En la actualidad existen múltiples metodologías de desarrollo de software, cada una de ellas más o menos útil para un tipo de proyecto u otro. En este proyecto se ha optado por una metodología ágil de tipo Scrum [8] debido a varios motivos. El primero de ellos es la naturaleza cambiante del proyecto. Se trata de un proyecto donde el resultado de una iteración puede afectar a las subsecuentes. Además, los conocimientos de los que se disponía eran limitados y se requería un proceso de aprendizaje en el ámbito de esta tecnología. Por otro parte, el extenso feedback proporcionado por los tutores semanalmente requería, en muchas ocasiones, de modificaciones en el software o nuevas funcionalidades. Por último, el tipo de entregas del TFG de carácter semanal/mensual coincidía con la filosofía de *Sprints* propuesta por Scrum.

4. Requerimientos del sistema

La primera tarea realizada en la fase de requerimientos ha sido la creación de una lista de requerimientos a cumplir del producto, conocida en Scrum como *Product Backlog*. Se trata de una lista dinámica que ha ido evolucionando a lo largo del proyecto. En la Fig. 1 se muestran el *Product Backlog* de los distintos bloques.

Bloque 1

Objetivos / Requerimientos	Priority	Status
Como usuario debo de poder seleccionar elementos con la mano y que estos aparezcan en dentro de otro elemento. **(Draggable and Droppable)**	High	Done
Como usuario debo de poder seleccionar elementos con la mano y pre visualizarlos en un entorno 3D. **(Viewable)**	High	Done
Como usuario debo de poder seleccionar elementos con la mano y moverlos libremente por pantalla. **(Movable)**	Medium	Done
Como usuario debo de poder moverme por secciones de la página con la mano. **(Visitable)**	High	Done
El sistema debe de permitir ejecutar funciones creadas por el usuario al detectar un clic con la mano sobre un elemento. **(Triggerable)**	High	Done
Como usuario debo de poder desplazar los elementos de un carrusel con la mano. **(Carrousel)**	High	Done

Bloque 2

Objetivos / Requerimientos	Priority	Status
Como usuario debo de poder reproducir, pausar y detener el sonido del audio.	High	Done
Como usuario debo de poder aumentar y disminuir el volumen del audio	High	Not Done
Como usuario debo de poder aumentar y disminuir la velocidad del audio	High	Done

Bloque 3

Objetivos / Requerimientos	Priority	Status
Como usuario debo de poder rotar la Perilla física y ver estos movimientos efectuados sobre el objeto en pantalla.	High	Done
Como usuario debo de poder subir y bajar el Slider y ver estos movimientos efectuados sobre el objeto en pantalla	Medium	Done
Como usuario debo de poder rotar la Manivela y ver estos movimientos efectuados sobre el objeto en pantalla	Low	Not Done
Desarrollo de un sistema que permita comparar el nivel de precisión de Leap Motion y el de un ratón en un entorno web.	Low	Done
Desarrollo de un sistema que permita comparar el nivel de velocidad de Leap Motion y el de un ratón en un entorno web.	Low	Done

Fig. 1 *Product Backlog del proyecto*

Desde las primeras fases del proyecto ya estaba claro, en mayor o menor medida, qué puntos se querían tratar. Así pues, el proyecto se ha dividido en tres grandes bloques, en los cuales se han realizado diversos Sprints de una o dos semanas.

En el primer bloque se ha llevado a cabo la creación de la Web sobre la que se han tratado los contenidos (tanto *back-end* como *front-end*) y se ha trabajado con contenidos multimedia en dos dimensiones (2D), haciendo uso de Leap Motion para mover el cursor como si de una pantalla táctil se tratase. Por otro lado, el segundo bloque se ha destinado a trabajar con contenidos de sonido. Finalmente, en el tercer y último bloque se ha trabajado en un entorno de tres dimensiones (3D) y se ha hecho uso de objetos del mundo real. La Fig. 2 muestra la planificación final del proyecto mediante un diagrama de Gantt, con la distribución en Sprints por semana de los bloques y después de haber sufrido varias modificaciones a lo largo del desarrollo.

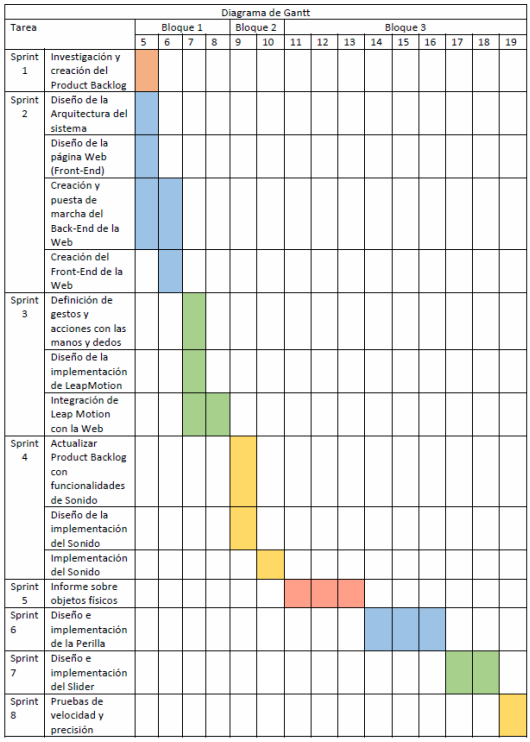


Fig. 2 *Diagrama de Gantt del proyecto*

Por otro lado, las reuniones han actuado como *Sprint Retrospective* y han servido para mejorar tanto el proyecto como la gestión de éste.

5. HERRAMIENTAS UTILIZADAS

Una vez definidos los requerimientos del sistema y profundizado en ellos, se ha llevado a cabo la selección del Software y Hardware a utilizar para la implementación del proyecto y la generación y mantenimiento de la documentación.

5.1. Software

5.1.1. Lenguajes informáticos

Se han utilizado los siguientes lenguajes de programación y marcado:

- JavaScript
- Python 2.7.11
- HTML y CSS

5.1.2. Programas, aplicaciones y librerías

Se han utilizado los siguientes programas y aplicaciones:

- Django (Back-end)
- Pip (Instalación de dependencias Django)
- Virtualenv (Entorno virtual para Django)
- Leap.js
- JQuery
- Bootstrap
- Flocking
- Leap Motion Controller
- Sublime Text 3
- Microsoft Office 2016
- Creately online
- Google Chrome
- Skype (Comunicación)

5.2. Hardware

Se ha utilizado la siguiente maquinaria y dispositivos:

- Ordenador personal
- Leap Motion
- Objetos físicos (Rosca, Slider y Manivela)

6. DISEÑO DEL SISTEMA

6.1. Creación del sitio web

Para llevar a cabo las pruebas de Leap Motion sobre los diferentes tipos de contenidos es necesario disponer primero de un entorno de trabajo. Las pruebas se han realizado en un entorno de navegador y, por lo tanto, ha sido necesaria la creación de una web muy básica donde poder llevarlas a cabo. Esta web se ha creado mediante Django [9], un framework basado en Python que ha permitido establecer de forma rápida y sencilla una estructura basada en el modelo-vista-controlador (MVC). También se ha utilizado Pip [10] para gestionar la instalación de dependencias de Django y Virtualenv [11] para la creación de un entorno virtual en el que ejecutar Django.

El primer diagrama realizado para entender el sistema como un conjunto ha sido el diagrama de la arquitectura. En él, se muestran los principales componentes del sistema y cómo interactúan entre ellos. Como se puede observar en la Fig. 3, el esquema muestra el modelo común de Cliente-Servidor donde el servidor atiende las peticiones del usuario y devuelve los recursos apropiados. En este caso, el servidor es ejecutado localmente (localhost). Entre los recursos HTML se distinguen diferentes vistas, una por cada tipo de prueba realizada. Así pues, las pruebas realizadas sobre los diferentes contenidos (objetos planos, sonido, objetos físicos, etc.) cuentan con su propia página web. Sucede lo mismo con los recursos JavaScript a excepción de la funcionalidad de los recursos planos, la cual ha sido integrada en el mismo archivo que sirve como interacción con Leap Motion. Otra observación a tener en cuenta es el intercambio de información que se produce entre el dispositivo Leap Motion y el archivo *leap.js* utilizando WebSockets. Se trata de un flujo constante de datos por el cual Leap Motion envía las coordenadas de las manos y dedos a este archivo.

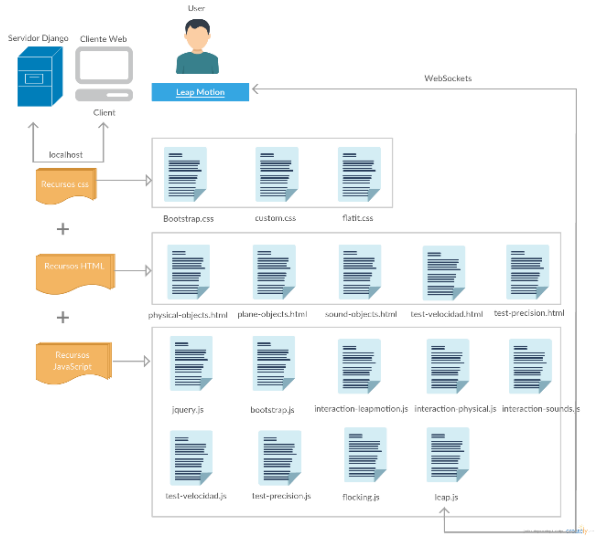


Fig. 3 *Diagrama de la arquitectura del sistema*

6.2. Diseño de los archivos JavaScript

Existen multitud de diagramas útiles a la hora de especificar y construir software. En este proyecto se ha optado por trabajar con dos tipos de diagramas:

- Diagramas de flujo.
- Diagramas de secuencia.

El diagrama de flujo es muy útil para mostrar, paso a paso, las operaciones que el sistema va realizando, así como las decisiones que toma para pasar de una operación a otra.

Por otro lado, el diagrama de secuencia muestra las interacciones que se realizan entre los diferentes componentes del sistema. Este diagrama se basa en el *Lenguaje unificado de modelado* (UML). Se trata de un estándar aprobado por la *Organización Internacional de Normalización* (ISO) y que cuenta con numerosos tipos de diagramas, cada uno ajustado a diferentes situaciones.

Cabe destacar que UML también cuenta con su propio diagrama de flujo conocido como diagrama de actividades. Sin embargo, se ha decidido no utilizar este diagrama ya que añadía demasiada complejidad y complicaba las cosas en comparación al simple.

A continuación, se mostrarán los diagramas utilizados a lo largo del proyecto para el diseño de las funcionalidades.

6.2.1. Diseño del bloque 1

El bloque 1 cuenta con un archivo JavaScript llamado *interaction-leapmotion.js* donde se implementa la interacción con Leap Motion, la visualización del esqueleto de la mano y el control de gestos y contenidos en 2D. Este archivo utiliza la librería de Leap Motion llamada *leap.js* e incluye el bucle principal del sistema. En la Fig. 4 se puede ver una iteración completa del bucle.

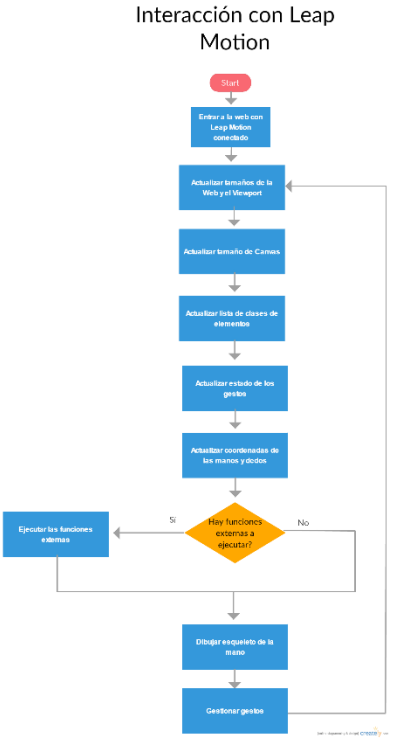


Fig. 4 *Diagrama de flujo del archivo interaction-leapmotion.js*

Como puede observarse, este archivo es capaz de ejecutar funciones externas. Esto es importante por la siguiente razón. Cada iteración del bucle puede verse como si de un frame se tratase. En cada frame se gestionan varios elementos como las coordenadas de manos y dedos, los gestos realizados por el usuario o los canvas HTML donde se dibujan las manos y dedos. Es importante, por tanto, que otros archivos JavaScript externos puedan ejecutar sus funcionalidades una vez por cada iteración, especialmente si quieren dibujar en el mismo canvas. Cabe destacar que los datos más importantes, como las coordenadas de las manos y dedos y el campo de visión del usuario (viewport), estan disponibles mediante variables globales, permitiendo así que otros archivos puedan utilizar esta información.

6.2.2. Diseño del bloque 2

En el bloque 2 se ha trabajado con contenidos multimedia de sonido. El diseño de este bloque se ha enfocado de diferente manera, siguiendo una programación basada en objetos. Si bien JavaScript no es un lenguaje orientado a objetos, existen técnicas para simular este comportamiento [12]. Para trabajar con sonidos se han creado diferentes módulos en el archivo *interaction-sounds.js* que facilitaban tareas diferentes. Algunos de estos objetos o módulos permiten la gestión de la velocidad del cursor, la creación de botones HTML o el propio control de canciones y sonidos

(pause, play, velocidad, etc.).

6.2.3. Diseño del bloque 3

El bloque 3 engloba todo lo relacionado con la interacción de contenidos físicos y se implementa en el archivo *interaction-physical.js*. El diseño sigue el mismo patrón que el del bloque 1, teniendo un bucle principal que llama a diversas funciones declaradas en el archivo. En la Fig. 5 puede observarse como este archivo interacciona con el resto de componentes. El archivo *interaction-physical.js* envía su módulo a *interaction-leapmotion.js* el cual lo ejecutará una vez por bucle.

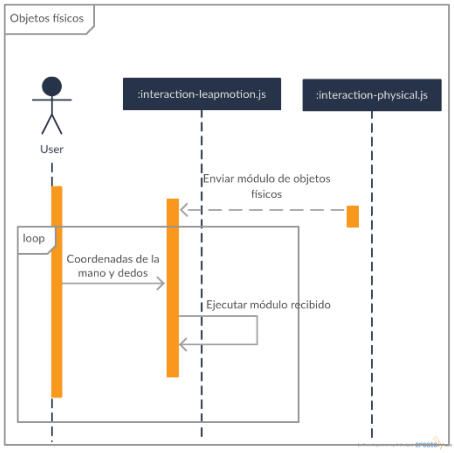


Fig. 5 *Diagrama de secuencia de interaction-physical.js*

En este bloque, además se han realizado algunas pruebas de velocidad y precisión para ver cómo se comporta Leap Motion en comparación a un ratón común. Estas pruebas se realizan en los archivos *test-velocidad.js* y *test-precision.js*. Tal y como puede observarse en la Fig. 6, el diagrama de secuencia sigue el mismo patrón de ejecución.

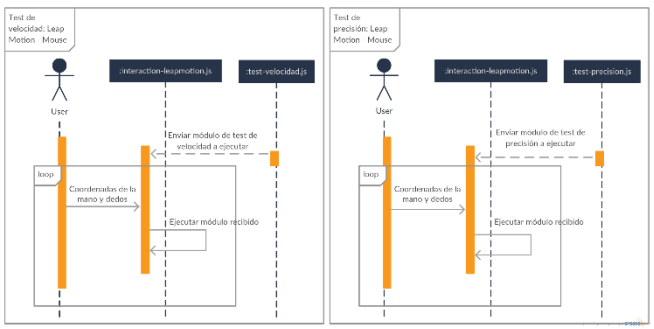


Fig. 6 *Diagramas de flujo de los tests de precisión y velocidad*

A parte, se han desarrollado dos diagramas de flujo (uno por test) que permiten entender mejor el proceso de decisiones que el software sigue para ejecutarlos. Las Fig. 7 y 8 muestra los diagramas de flujo de ambos tests.

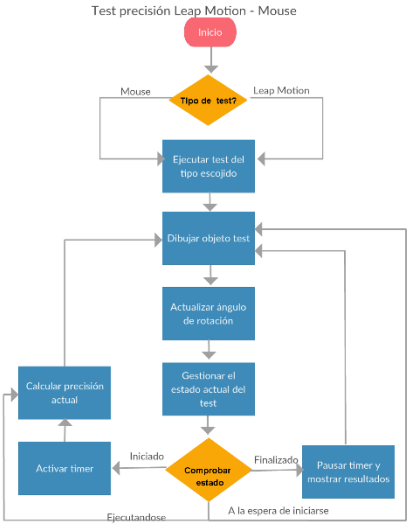


Fig. 7 *Diagrama de flujo del test de precisión*

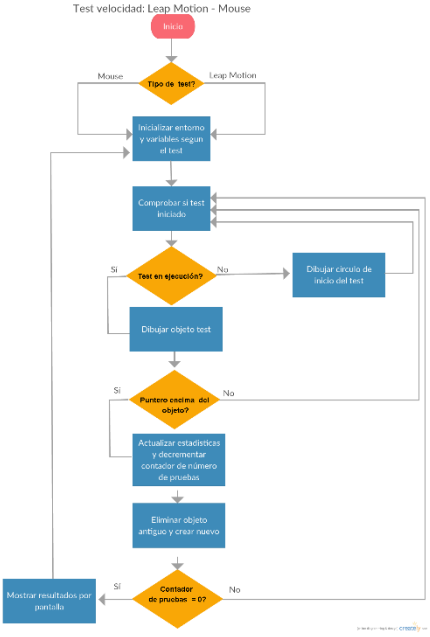


Fig. 8 *Diagrama de flujo del test de velocidad*

7. PRUEBAS DEL SISTEMA

La fase de testeo permite asegurarse de que la calidad del software se encuentra por encima de las métricas establecidas y que este cumple con las especificaciones estipuladas en la fase de requerimientos. Durante la realización de este proyecto se han llevado a cabo dos tipos de pruebas:

- Pruebas de sistema. Permiten comprobar que el sistema funciona tal y como el usuario/cliente espera.
- Pruebas de Regresión. Permiten verificar que todo el software previamente realizado sigue funcionando.

Las aplicaciones realizadas en JavaScript mezclan, en

muchas ocasiones, lógica de backend y elementos HTML. Este proyecto es uno de esos casos. Muchos de los resultados de ejecutar ciertas funciones son cambios en la posición de elementos, tamaños, etc. Por otro lado, los sistemas creados no tienen el objetivo de ser comercializados y sirven únicamente como entorno para realizar pruebas sobre diferentes tipos de contenidos. Es por ello, que no se han incluido tests unitarios y se ha optado directamente por realizar pruebas de sistema.

Las pruebas de regresión se han realizado cada vez que se completaba un requerimiento del *Product Backlog* para asegurar que todo seguía funcionando correctamente

8. Análisis de precisión y velocidad del dispositivo

8.1. Pruebas

Se ha realizado un análisis de la velocidad y precisión de Leap Motion a la hora de interactuar con manos y dedos. El objetivo era comprender un poco mejor las limitaciones o puntos fuertes del dispositivo en comparación a un ratón común. Pese a que estos tests se han realizado durante el bloque 3, lo ideal hubiese sido realizarlos al inicio del proyecto.

El primer test realizado ha sido el de precisión. En este test, el usuario debía seguir con el dedo índice el perímetro de un circulo dibujado en pantalla, ajustándose a la línea lo máximo posible. Este test devuelve la precisión del usuario en forma de porcentaje. Para ello, el test hace comprobaciones de la distancia a la que el dedo se encuentra del centro del círculo y la compara con el área de éste. Este mismo test se ha repetido con el usuario haciendo uso de un ratón.

El segundo test realizado ha sido el de velocidad. En él, el usuario debía situar el dedo índice sobre los elementos que iban apareciendo en pantalla hasta señalar 10 de ellos. El test devolvía el tiempo que ha tardado el usuario en completarlo.

8.2. Resultados

Los resultados del test de precisión después de 44 ejecuciones pueden observarse en la Fig. 9. En los resultados puede verse como ambos periféricos siguen un patrón común. Tiempos más altos permiten al usuario ser más preciso. Sin embargo, la precisión del ratón en tiempos similares es algo superior a la de Leap Motion.

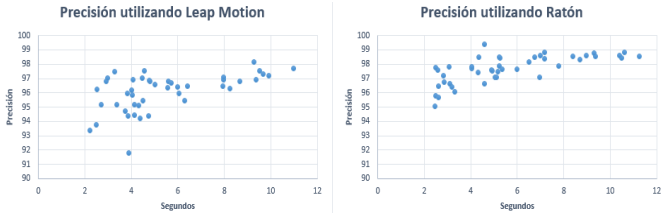


Fig. 9 *Gráficas con los tiempos y precisión de cada periférico*

Por otro lado, en la Fig. 10 pueden verse los resultados del test de velocidad. El test muestra como la velocidad de interacción utilizando un ratón es muy superior a la de Leap Motion.

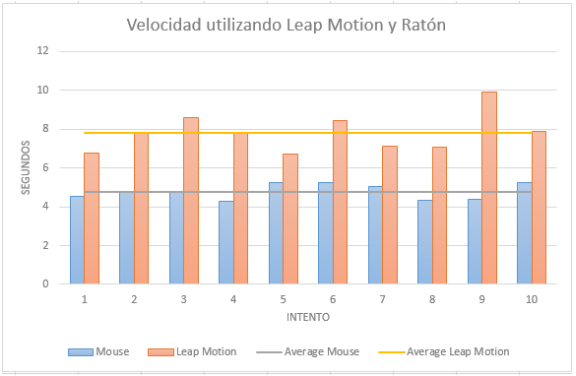


Fig. 10 *Gráfica con el tiempo tardado en completar cada test según el periférico*

Como último punto, cabe destacar que los tests dependen en gran medida de quien los realiza, ya que puede haber usuarios que sean más rápidos o tengas más precisión que otros.

9. Bloque 1: Interacción con contenidos planos

El primer bloque del proyecto ha consistido en un conjunto de pruebas sobre contenidos en dos dimensiones. En este bloque, Leap Motion actúa como si de una pantalla táctil se tratase, traduciendo las coordenadas de manos y dedos sobre un plano con coordenadas (x, y). La API JavaScript de Leap Motion viene con soporte de detección de gestos como Swipe, Tap o movimientos circulares [13].

9.1. Pruebas

Se han creado un total de 6 tipos de objetos y acciones para las pruebas, tal y como consta en el *Product Backlog*. Cada objeto y acción tiene un código único y basta con añadir este código a la propiedad "class" del elemento HTML sobre el que queremos que la acción se lleve a cabo. La Fig. 11 muestra todas ellas junto a su código.

Acción	Código
Drag and drop	*leap-draggable* y *leap-droppable*
Visualizar un elemento	*leap-viewable*
Mover un elemento	*leap-movable*
Navegar por la página actual	*leap-visitable*
Acción custom	*leap-triggerable*
Hacer girar el Carrousel	*carrousel* y *carrousel-container*

Fig. 11 *Tipos de objetos y acciones para trabajar con contenidos en dos dimensiones*

La posición de Leap Motion utilizada en estas pruebas es la estándar, situando el dispositivo en frente de la pantalla y con los sensores apuntando hacia arriba.

Las pruebas realizadas con estos elementos se muestran en la sección del Apéndice 1.

9.2. Resultados

Los resultados han sido muy positivos. Leap Motion interactúa realmente bien con este tipo de contenidos. A continuación, algunas de las observaciones al trabajar con estos contenidos:

- El dispositivo es menos preciso que un ratón. A esto hay que sumarle también el temblor de las manos que, pese a ser mínimo, repercute en la experiencia al trabajar con contenidos multimedia de tamaño reducido.
- Leap Motion trabaja especialmente bien cuando se utilizan gestos o valores discretos. Utilizando valores discretos se filtran pequeños movimientos involuntarios.
- El espacio que el dispositivo proporciona al usuario para interactuar es relativamente reducido.
- El uso del dispositivo provoca cansancio en los brazos y, por lo tanto, es conveniente usarlo para acciones de corta duración como, por ejemplo, hacer girar un *carrousel*.
- Leap Motion es muy útil para rotar contenidos. Puede ser práctico en tiendas online permitiendo un mejor control a la hora de visualizar productos.

10. Bloque 2: Interacción con contenidos de sonido

En el bloque 2 se ha trabajado con sonidos. Para ello, se hace uso de una librería externa llamada Flocking que ofrece herramientas para trabajar con sonidos.

10.1. Pruebas

Se han realizado pruebas con sonidos y música. En ellas se ha utilizado Leap Motion para pausar o reproducir música, aumentar la frecuencia de los sonidos, etc. Además, se han explorado otras posibilidades de uso del dispositivo en este campo.

En el apartado del Apéndice 2 pueden verse algunas de estas pruebas.

10.2. Resultados

Los resultados en este campo han sido mixtos. A continuación, se hace un repaso de algunas observaciones extraídas del trabajo con estos contenidos:

- En un campo como el de la música se requiere mucha precisión para controlar aspectos como el volumen o ritmo. Leap Motion no aporta este nivel de precisión.
- Los valores discretos pueden ayudar a trabajar con este tipo de contenidos.
- Leap Motion abre muchas posibilidades en el campo de la música. Se pueden crear instrumentos y utilizar Leap Motion para practicar con ellos.

11. Bloque 3: Interacción con contenidos físicos

En el tercer y último bloque se ha tratado de explorar nuevas formas de utilizar el dispositivo. Tras varias reuniones presenciales y a distancia con los tutores, se planteó la idea de utilizar Leap Motion como dispositivo para interactuar con objetos del mundo real. El planteamiento puede parecer extraño en un principio debido a que Leap Motion no está preparado para este tipo de interacciones, pero las ventajas son realmente prometedoras:

- Feedback visual. Los usuarios conocen constantemente la posición espacial de los objetos. Además, estos actúan como *Esqueumorfismos* y permiten al usuario saber cómo funciona cada uno de ellos y que limitaciones tienen [14].
- Puro software. El objeto con el que interactúan los usuarios no necesita estar conectado a ninguna máquina de manera física. Los usuarios pueden interactuar con botones, sliders, manivelas o cualquier clase de dispositivo siendo el software el único encargado de ello.

11.1. Informe previo

El primer paso que se ha dado para trabajar con este tipo de contenidos ha sido el de realizar un estudio previo en el que se ha analizado el grado de compatibilidad de Leap Motion con objetos físicos. El sensor de Leap Motion está preparado para detectar la forma de las manos y los dedos. Sin embargo, al tocar objetos con la mano se puede modificar la forma de ésta, haciendo que Leap Motion no sea capaz de detectarla. Por otro lado, hay que tener en cuenta que el hecho de utilizar objetos físicos puede afectar al campo de visión de Leap Motion obstruyendo su visión. En el Apéndice 3 se incluyen todas las pruebas realizadas para la creación de este informe. Las pruebas se han llevan a cabo con 3 tipos de objetos: rosca, slider y manivela.

11.2. Pruebas

Una vez realizado el informe y profundizado en las limitaciones del dispositivo, se decidió un orden para implementar cada uno de los objetos. Se ha priorizado el trabajo con la rosca y el slider ya que han sido los elementos que mejores resultados han dado en el informe. Para trabajar con estos objetos en un entorno en tres dimensiones es muy importante tener un control completo de las manos y los dedos. Es por ello que, antes de implementar el software de control de los objetos, se introdujo de forma previa un sistema de visualización que permite ver la mano desde diferentes ángulos y situar el dispositivo Leap Motion en diferentes posiciones. La Fig. 12 muestra el esqueleto de la mano desde diferentes ángulos.

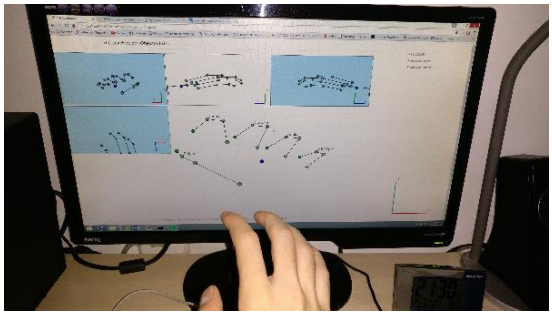


Fig. 12 *Esqueleto de la mano desde diferentes ángulos*

En la Fig. 13 puede verse Leap Motion situado en diferentes posiciones. Gracias a este sistema se puede cubrir un rango de posiciones de la mano y objetos más amplio, adaptándose mejor a la situación y evitando que Leap Motion pierda el ángulo de visión con la mano.

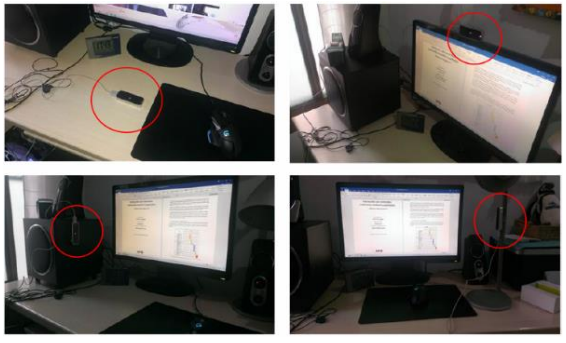


Fig. 13 *Leap Motion situado en diferentes posiciones*

11.2.1. Rosca

El primer controlador implementado durante el proyecto es el de la rosca. Se entiende como rosca el mecanismo de control rotable que permite graduar otros dispositivos. El controlador permite al usuario interactuar con una rosca rotándola a derecha e izquierda. Para su implementación, se han utilizado varios algoritmos y se han tenido en cuenta varios factores.

Lo primero ha sido establecer un área de interacción esférica entorno al objeto. Esto quiere decir que el objeto solo funcionará si la palma del usuario se encuentra dentro de esta área. La elección de la palma como punto de control se debe a la frecuente pérdida de visión y coordenadas de los dedos. El algoritmo que comprueba si la palma se encuentra dentro del área de interacción se basa en la fórmula de la distancia Euclidiana de la Fig. 14. De esta manera se calcula la distancia entre la palma y el centro de la rosca. Si esta distancia es superior a una cota arbitraria establecida la rosca no funcionará.

$$d(\mathbf{x},\mathbf{y}) = \|\mathbf{x}-\mathbf{y}\| = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Fig. 14 *Distancia Euclidiana en el espacio Rn*

En segundo lugar, se comprueba que los dedos realmente están sujetando la rosca. Para ello, se calcula el perímetro del triángulo formado por los dedos pulgar, índice y corazón, tal y como se muestra en la Fig. 15. Si el perímetro es superior a la cota establecida la interacción no funcionará, pues el sistema supondrá que el usuario no está agarrando la rosca.

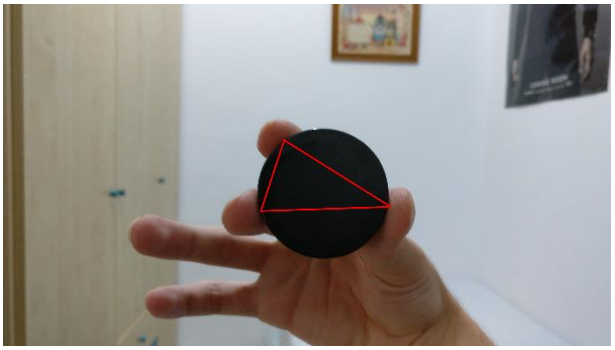


Fig. 15 *Triángulo formado por los dedos pulgar, índice y corazón*

El tercer punto a tener en cuenta es la perdida puntual de la coordenada de los dedos y los movimientos bruscos que estos sufren en ocasiones. Para reducir este problema se han aplicado dos mecanismos. El primero es un mecanismo de suavizado de rotación. Las rotaciones muy bruscas en la roscs se filtran haciendo que la rotación se produzca a lo largo de un intervalo de tiempo. El segundo mecanismo se basa en un período de gracia. Cuando se deja de detectar la interacción con la rosca, el sistema permite al usuario seguir usándola durante un breve periodo de tiempo, evitando así que la perdida puntual de coordenadas no afecte a la experiencia del usuario.

En el Apéndice 4 se incluyen muestras de la interacción con la rosca.

11.2.2. Slider

El segundo controlador desarrollado ha sido el del slider. Por slider se entiende el mecanismo que permite desplazar horizontal o verticalmente un indicador a través de un rail.

El objetivo era utilizar los mismos algoritmos que en la rosca adaptándolos al slider. Sin embargo, por razones de tiempo, no se ha implementado ningún tipo de suavizado ni periodo de gracia en este último.

En el Apéndice 5 se incluyen muestras de como funciona la interacción con el slider.

11.2.3. Manivela

No ha sido posible realizar el controlador de la manivela por razones de tiempo y complejidad. Se trata del elemento que más problemas producía al rastrear sus coordenadas.

Las pruebas realizadas sobre la manivela en el informe previo mostraban como, durante la interacción, Leap Motion cambiaba las coordenadas de manos y dedos bruscamente, haciendo que se desplazasen por pantalla de una manera que no se correspondía a la realidad. Había momentos en los que incluso se perdía la mano por completo y, por lo tanto, no se disponía de coordenadas con las que trabajar. Esto es debido a que Leap Motion no es capaz de distinguir la mano cuando ésta se junta con la manivela.

11.3. Resultados

A pesar de las limitaciones del dispositivo, los resultados han sido positivos. Pese a lo poco prometedoras que eran las pruebas en el informe previo de los objetos, el uso de varias técnicas y algoritmos ha mejorado en gran medida la experiencia del usuario al interactuar con los objetos.

La principal limitación al trabajar con este tipo de contenidos es la perdida de forma de la mano y la obstrucción de visión que el dispositivo sufre en ocasiones. Hubiese sido interesante poder utilizar varios dispositivos Leap Motion funcionando en paralelo y situados en diferentes posiciones, cubriendo así diferentes ángulos de visión.

12. Conclusiones

A lo largo del proyecto se ha trabajado con todo tipo de contenidos con el objetivo de mostrar al lector las posibilidades que ofrece un dispositivo de detección de movimientos como Leap Motion. Como se ha podido observar, las opciones que ofrece el dispositivo son muy amplias y variadas.

En cuanto a las características del dispositivo, destacan la disponibilidad de una nueva dimensión del espacio en la que trabajar y el reconocimiento de gestos, siendo estos últimos muy útiles en acciones poco complejas. Si bien el dispositivo presenta ventajas respecto a otras formas de interacción más comunes como el ratón, no hay que olvidarse de sus limitaciones. El dispositivo puede tener problemas a la hora de mantener contacto visual con las manos, necesita unas condiciones de entorno óptimas y es claramente menos preciso y rápido que un ratón.

Entre las diferentes pruebas, Leap Motion ha hecho posible la creación de una serie de controladores que permiten la interacción con objetos cotidianos. Los resultados obtenidos con la rosca y el slider han sido positivos, pero podrían haber mejorado aplicando nuevos algoritmos o haciendo uso de múltiples Leap Motion's.

Por último, cabe destacar que tan solo se han realizado los controladores de 2 de los 3 objetos planeados en el tercer bloque, quedando fuera la manivela. Aspectos a tener en cuenta en futuras extensiones del trabajo son la realización de unas pruebas de test más completas y el uso de múltiples dispositivos Leap Motion para obtener mejores resultados.

Agradecimientos

Quiero aprovechar esta sección para agradecer a mis tutores del proyecto, Fernando Luis Vilariño Freire y Daniel Alexander Norton, el soporte prestado en la realización de este proyecto.

Bibliografía

[1] E. Butow, User Interface Design for Mere Mortals. Pearson Education, 2007, Chapter 2.

[2] M. Mihelj, D. Novak and S. Beguš, Virtual Reality Technology and Applications. Springer Science & Business Media, 2013, pp. 7-10.

[3] A. Sears and J. Jacko, The Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies and Emerging Applications, 2nd ed. CRC Press, 2007, p. 623-625.

[4] T. Parisi, Programming 3D Applications with HTML5 and WebGL: 3D Animation and Visualization for Web Pages. O'Reilly Media, Inc, 2014, p. 17.

[5] L. E. Christine, L. Carter, J. Araullo, The Leap Motion controller: a view on sign language, 2013. [Online]. Available: https://www.researchgate.net/profile/Jake_Araullo/publication/262172749_The_Leap_Motion_controller_a_view_on_sign_language/links/566e153008ae1a797e405edf.pdf

[6] D. Bassily, C. Georgoulas, J. Güttler, T. Linner and T. Bock, "Intuitive and Adaptive Robotic Arm Manipulation using the Leap Motion Controller", Conference ISR ROBOTIK 2014, 2014.

[7] D. Bachmann, F. Weichert and G. Rinkenauer, Evaluation of the Leap Motion Controller as a New Contact-Free Pointing Device, 2014. [Online]. Available: http://www.mdpi.com/1424-8220/15/1/214/html [Accessed: 23- Jun- 2016]

[8] K. Schwaber and J. Sutherland, "The Scrum Guide", Scrumguides.org, 2013. [Online]. Available: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf. [Accessed: 23- Jun- 2016].

[9] "Django documentation | Django documentation | Django", Docs.djangoproject.com, 2016. [Online]. Available: https://docs.djangoproject.com/en/1.9/. [Accessed: 23- Jun- 2016].

[10] "pip — pip 8.1.2 documentation", Pip.pypa.io, 2016. [Online]. Available: https://pip.pypa.io/en/stable/. [Accessed: 24- Jun- 2016].

[11] "Virtualenv — virtualenv 15.0.2 documentation", Virtualenv.pypa.io, 2016. [Online]. Available: https://virtualenv.pypa.io/en/stable/. [Accessed: 24- Jun- 2016].

[12] "Introducción a JavaScript orientado a objetos", Mozilla Developer Network, 2016. [Online]. Available: https://developer.mozilla.org/es/docs/Web/JavaScript/Introducci%C3%B3n_a_JavaScript_orientado_a_objetos. [Accessed: 23- Jun- 2016].

[13] "JavaScript SDK Documentation — Leap Motion JavaScript SDK v3.1 documentation", Developer.leapmotion.com, 2016. [Online]. Available: https://developer.leapmotion.com/documentation/javascript/index.html. [Accessed: 23- Jun- 2016].

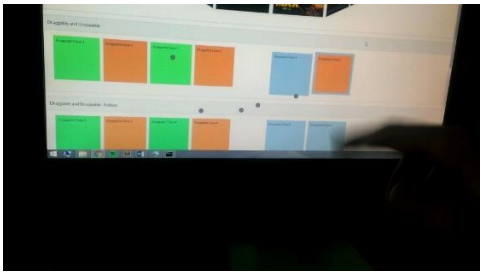
[14] A. Marcus, "Design, User Experience, and Usability: User Experience Design for Diverse Interaction Platforms and Environments. Part 2", Third International Conference, DUXU 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, 2014, p. 363.

- *E-mail de contacte: daniel.garciap@e-campus.uab.cat*
- *Menció realitzada: Enginyeria del Software*
- *Treball tutoritzat per: Fernando Luis Vilariño Freire (CVC) y Daniel Alexander Norton*
- *Curs 2015/16*

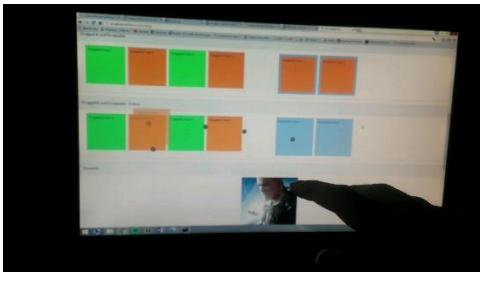
APÉNDICE

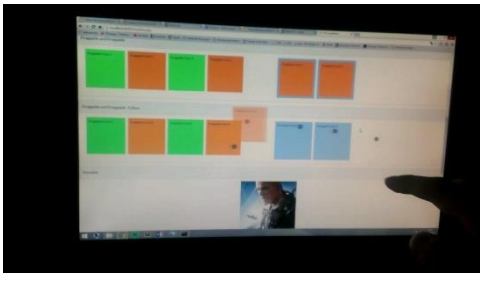
1. Interacción con contenidos planos

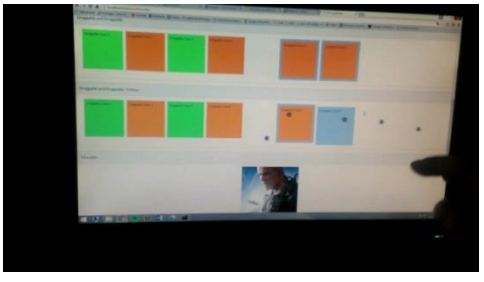
1. *Drag and Drop*



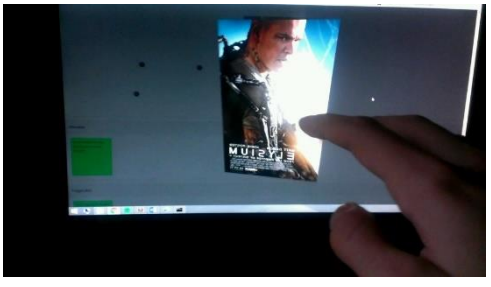
2. *Drag and Drop - Follow*



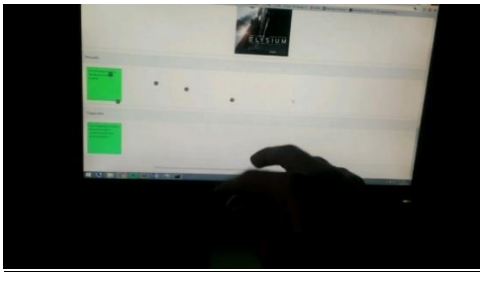


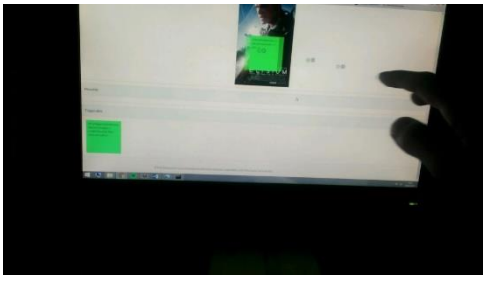


3. *Visualización de elementos*

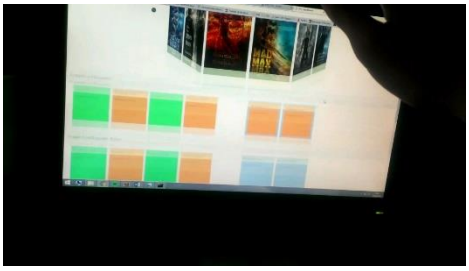


4. *Desplazamiento de elementos*

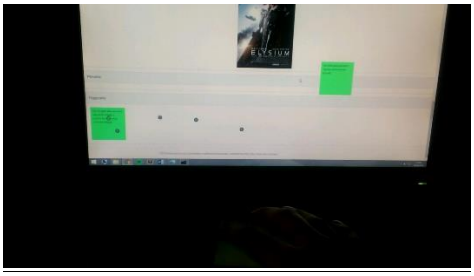
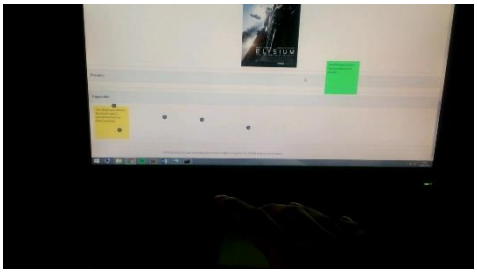




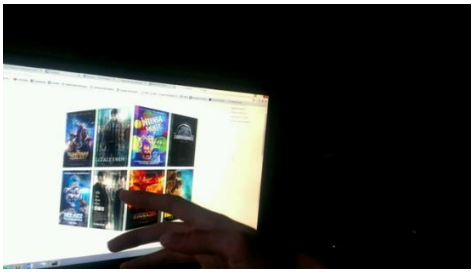
5. *Nevegación por la web*

6. *Soporte de ejecución de funciones externas. (En el ejemplo, modificar color del elemento)*

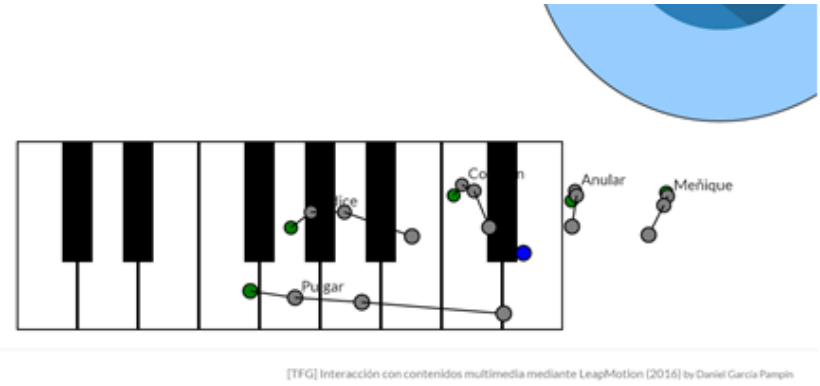



7. *Rotación de Carrousel*



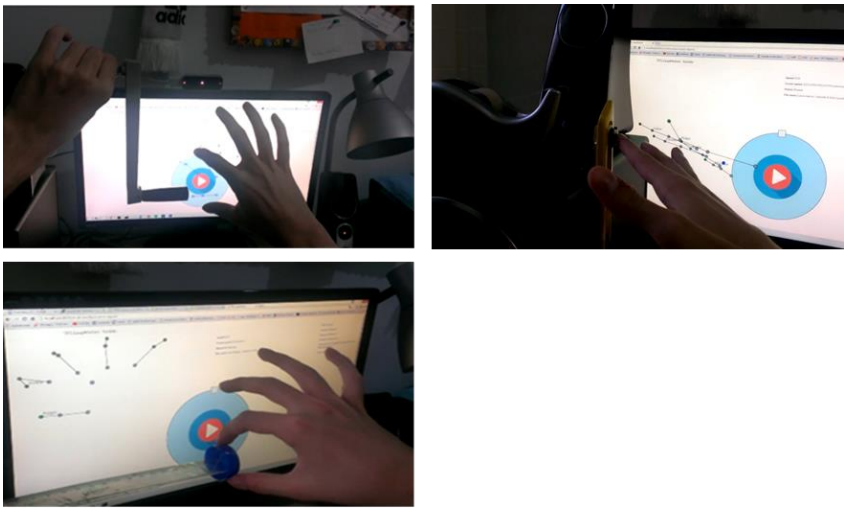
2. Interacción con contenidos de sonido

8. *Pruebas con sonido. La frecuencia del sonido depende de la velocidad a la que se mueve el dedo índice*



***9**. Posibles usos de Leap Motion en el ámbito musical. En el ejemplo, tocando un piano con Leap Motion*

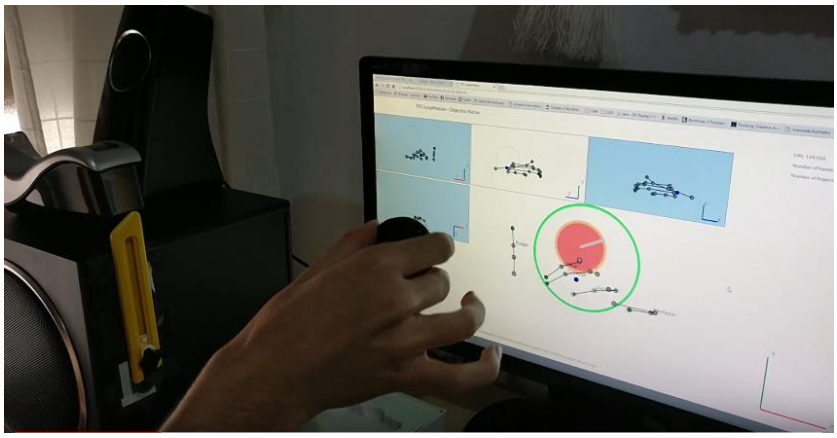
3. Informe de contenidos físicos



***10**. Pruebas realizadas para el informe previo al trabajo con objetos físicos*

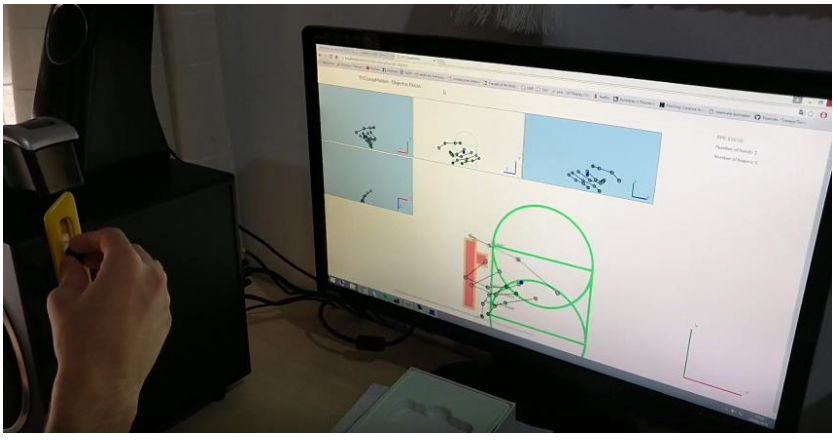
4. Interacción con contenidos físicos: Rosca

***11**. Rotando la rosca hacia la izquierda. El movimiento se traduce en la pantalla. El círculo verde del monitor es el área de interacción*

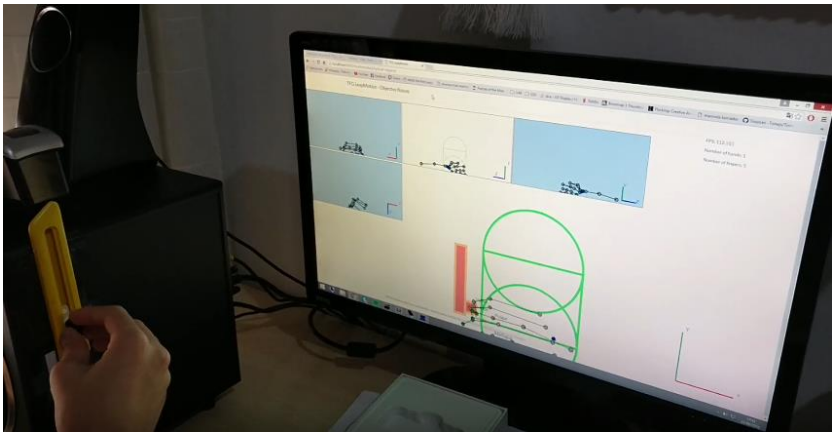


***12**. Rotación hacia la derecha*

5. Interacción con contenidos físicos: Slider



***13**. Desplazando el slider hacia arriba. El área verde del monitor es el área de interacción*



***14**. Desplazando el slider hacia abajo*

En el siguinte enlace hay disponible un conjunto de vídeos mostrando en funcionamiento el dispositivo en las diferentes partes del proyecto:

https://www.youtube.com/playlist?list=PLFfDYrYc880fp7xqqmhtTDcCCne_ZKoxD