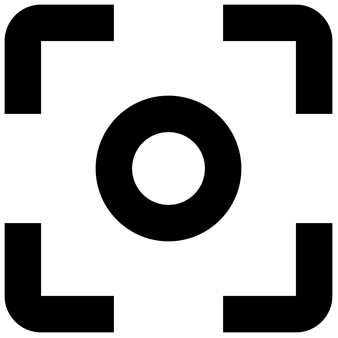
PERSPECTIVE

Cuestión de perspectiva

Memoria del proyecto



UNIVERSIDAD COMPLUTENSE
MADRID

Facultad
de
Informática

Participantes
Iván Aguilera Calle
Daniel García Moreno
David Llop Vila
Fernando Viñas Martins

Grupo 05

Máster en Ingeniería Informática - UCM - FDI
Tecnologías Multimedia e Interacción
2017 - 2018

1. Introducción

A través de la redacción del presente documento, se pretende reflejar el trabajo realizado, la funcionalidad implementada, los problemas y otros puntos relevantes surgidos durante la elaboración del proyecto “Perspective”, el cual se encuentra enmarcado en la asignatura de Tecnologías Multimedia e Interacción (TMI), impartida durante el curso 2017/2018 en la Facultad de Informática de la Universidad Complutense de Madrid.

El proyecto, con nombre “Perspective” ha sido desarrollado de manera colaborativa y apoyándose en herramientas de control de versiones, como GitHub, por el Grupo 05 del máster en Ingeniería Informática, formado por:

- Iván Aguilera Calle
- Daniel García Moreno
- David Llop Vila
- Fernando Viñas Martins

2. Motivación

Este proyecto plantea como iniciativa el desarrollo de una aplicación multimedia que permita la corrección dinámica de la perspectiva de las imágenes captadas por una webcam. Como núcleo medular se busca conseguir la corrección de la perspectiva de una imagen.

El proyecto otorgará la posibilidad a los usuarios de interactuar con la perspectiva de un vídeo, buscando tanto aprender como dejar reflejado aquello se puede conseguir al hacer un empleo combinado de una serie de diversos conocimientos matemáticos y tecnológicos.

Las principales utilidades que hemos encontrado entre otras son:

- Corrección de perspectiva de vídeo en las conferencias.
- Corrección de perspectiva en escaneo de documentos (tickets).
- Visualización de un objeto con la perspectiva correcta: por ejemplo, de una Pizarra girada, ponerla de frente.
- Modificación de la perspectiva de la imagen en función de los deseos del usuario.
- Medición de distancias en imágenes tomadas por cámaras de tráfico.
- Contabilización de vehículos en un determinado tramo de carretera.

Los resultados que se pueden conseguir son muy potentes y diversos. Además, tenemos la intención de conseguir más y nuevos resultados de manera incremental con el objetivo de investigar y seguir probando en combinación otras herramientas.

3. Descripción

El elemento de perspectiva es utilizado para hacer referencia a la dimensión de los objetos en una imagen y a la relación espacial que existe entre ellos respecto a un punto de vista. Jugando con la perspectiva se consiguen efectos sorprendentes aplicando distintas transformaciones.

La transformación o distorsión de la perspectiva de una imagen es la transformación propiamente dicha de un objeto y el área que lo rodea. Esto provoca que el aspecto del conjunto sea distinto respecto a cómo se vería presentado una longitud focal normal. Por lo tanto, juega un papel muy importante el ángulo de visión. Con este proyecto se quiere jugar con el ángulo de visión simulando que una fotografía ha sido, por ejemplo, tomada desde otro punto de vista, como se visualiza en el siguiente ejemplo.

A continuación podemos observar cómo transformamos la perspectiva de una fotografía de la facultad para poder ver las letras de frente. De la misma manera, vemos como otras zonas de la foto quedan deformadas, como es el caso de las ventanas.

Imagen original

Parte de frente seleccionada

Imagen con perspectiva transformada

Nuestra aplicación captará vídeo a partir de una webcam o de una cámara de teléfono móvil y mostrará las imágenes en la interfaz web. El usuario podrá seleccionar cuatro puntos de esa imágen y el programa reajustará la perspectiva de la imágen para que esta se visualice desde esa perspectiva. Otras funcionalidades paralelas es la de poder invertir una imagen o poder realizar un efecto “espejo” de la misma.

4. Perspective en funcionamiento

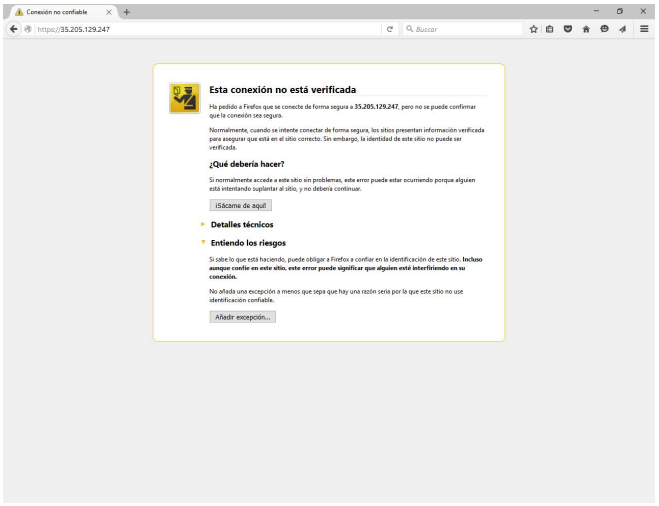
En el presente apartado se explica cómo podemos bajarnos el proyecto Perspective para realizar la instalación y para poder acceder a la aplicación web que contiene la funcionalidad desarrollada.

4.1. Acceso a la aplicación en producción

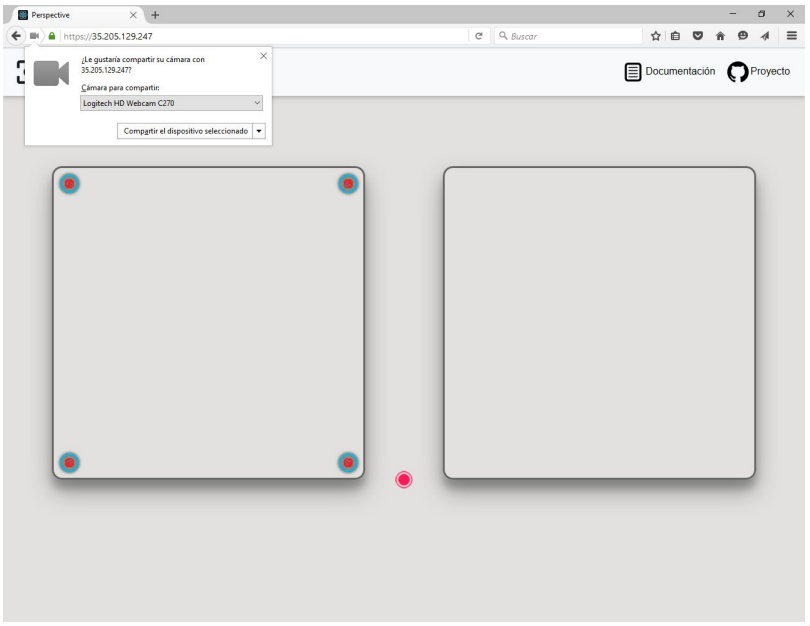
Como requisito para la entrega del proyecto, se ha procedido a la instalación de la aplicación en un servidor remoto, con el fin de facilitar la evaluación del proyecto. El servicio que hemos elegido para este fin es: Google Cloud Platform, y si se accede a la siguiente dirección se puede empezar a utilizar la aplicación directamente sin tener que realizar la instalación: https://35.205.129.247

Recalcar la importancia de utilizar la url anteriormente con https (con s), si no el enlace no funciona. Nada más introducir la URL, aparecerá un mensaje del navegador indican que el sitio no es seguro (esto se debe a que no utilizamos ningún certificado de confianza https). Se deberá dar a continuar (dependiendo del navegador saldrá un mensaje u otro, usar Google Chrome para un correcto funcionamiento de la aplicación web).

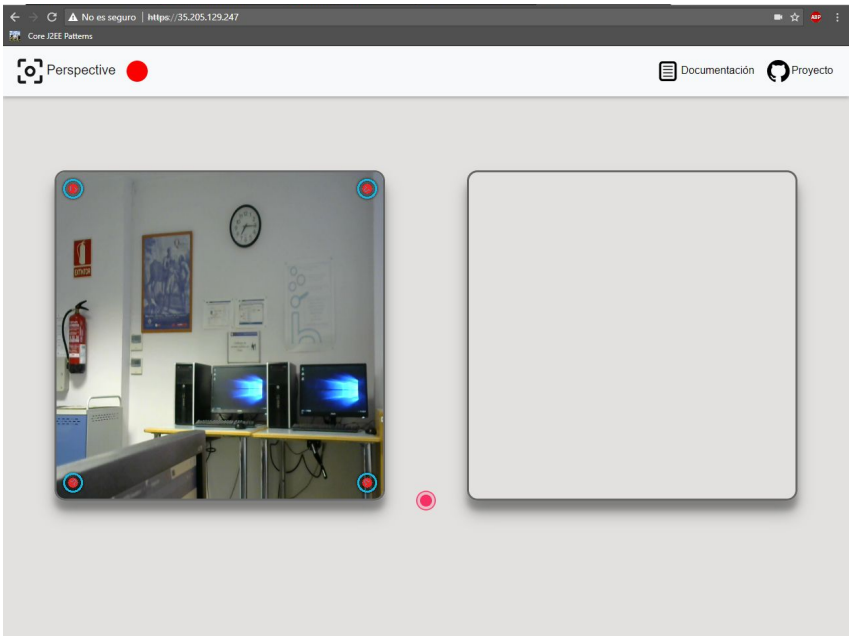
- Perspective:



Continuar (depende del navegador sale un aviso de una forma u otra), y aparecerá lo siguiente:



Para poder utilizar la aplicación, es necesario dar permiso para activar la webcam (para que funcione la webcam, acceder a la aplicación web desde Google Chrome).



4.2. Instalación

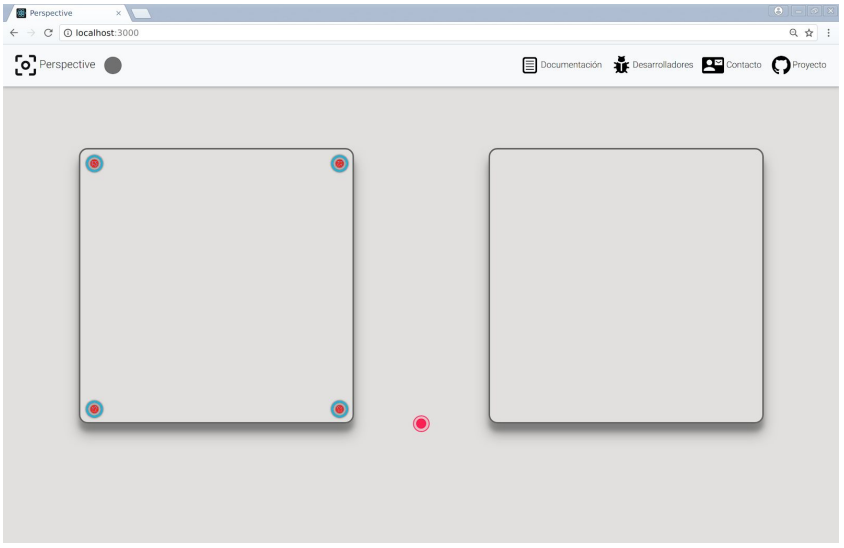
El proceso por el cual se realiza y se arranca la instalación del proyecto "Perspective" es muy simple y sencillo, solamente hay que realizar los siguientes pasos:

1. Clonar el repositorio del proyecto en alguna directorio: https://github.com/11crom11/Perspective.git
    a. Para que funcione el proyecto es necesario instalar NodeJS: https://nodejs.org/es/.
2. Entrar a la carpeta Perspective/Perspective
3. Ejecutar comando npm install para instalar las dependencias del React
4. Ejecutar comando npm start para iniciar la aplicación Perspective
5. Una vez realizados los pasos anteriores, ya podemos acceder al navegador y empezar a utilizar la aplicación Perspective localhost:3000 o Hostname:3000

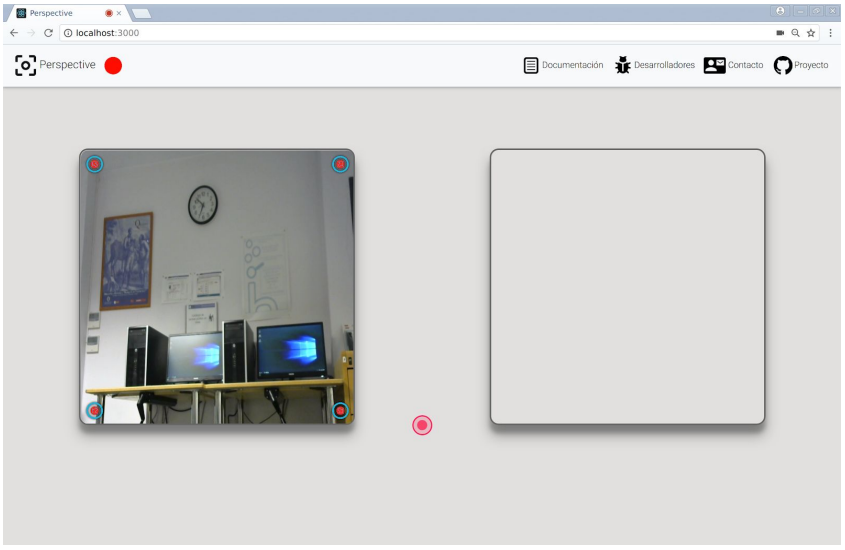
4.3. Manual de uso

En el siguiente apartado se detallará la forma de utilizar la aplicación web desarrollada para el proyecto.

Una vez arrancado la aplicación web y accedido a su respectiva url, la primera imagen que obtenemos es la siguiente:

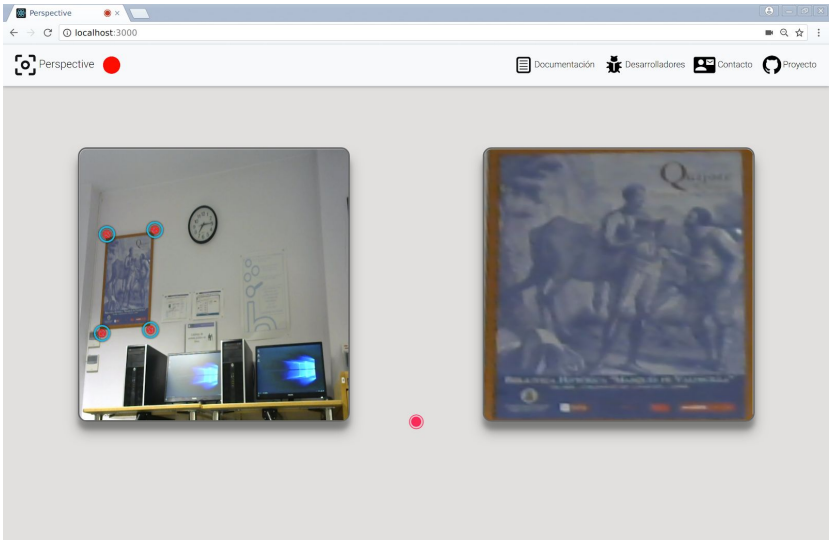


En la anterior imagen podemos observar la pantalla principal de la aplicación web. El cuadrado de la izquierda muestra el vídeo que actualmente está la webcam capturando, y en el recuadro de la derecha se muestra la imagen transformada del área que el usuario ha seleccionado. Arriba tenemos la barra de navegación donde figuran las diferentes secciones de la aplicación web. En el lado izquierdo podemos observar un punto de color gris que nos indica el estado de la webcam. El color gris simboliza que la cámara está desactivada y debemos activarla. Para ello, debemos dar permiso desde el navegador y refrescar la página. Una vez activada la webcam, el estado de la ventana principal sería el siguiente:



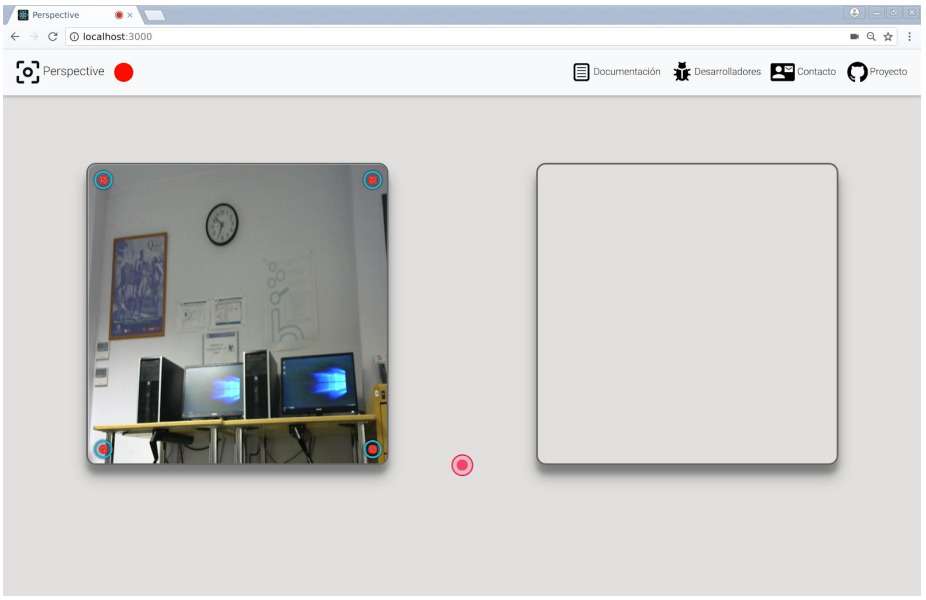
Como se puede observar, el color del círculo superior izquierda ha pasado de un tono gris a un tono rojo y ahora, en el cuadrado de la izquierda se visualiza imagen.

A partir de aquí, ya podemos obtener una imagen transformada en tiempo real desplazando los cuatro puntos colocados en las esquinas del cuadrado de la izquierda. Si por ejemplo, queremos cambiar la perspectiva del cuadro colocado en la pared de la imagen, a la izquierda del reloj, debemos colocar los puntos en las esquinas del cuadro, obteniendo el siguiente resultado:

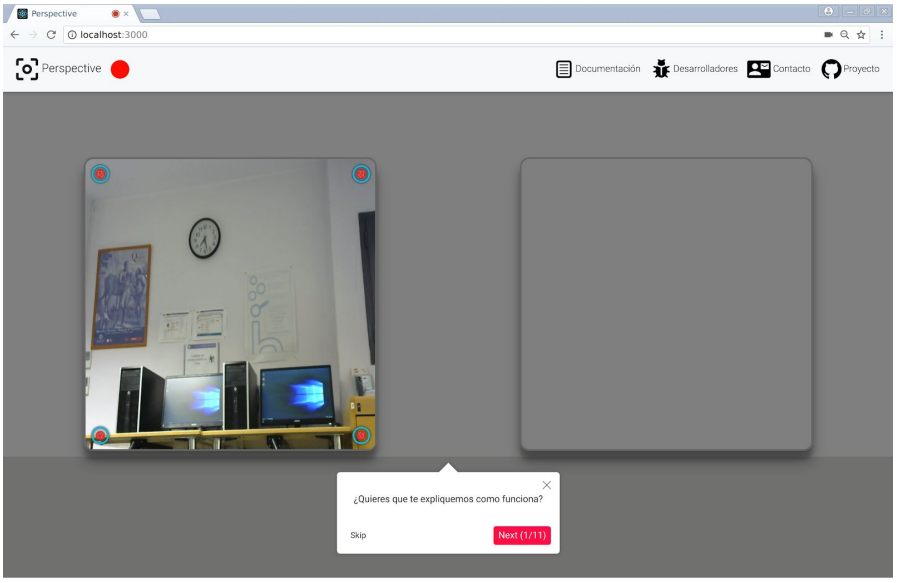


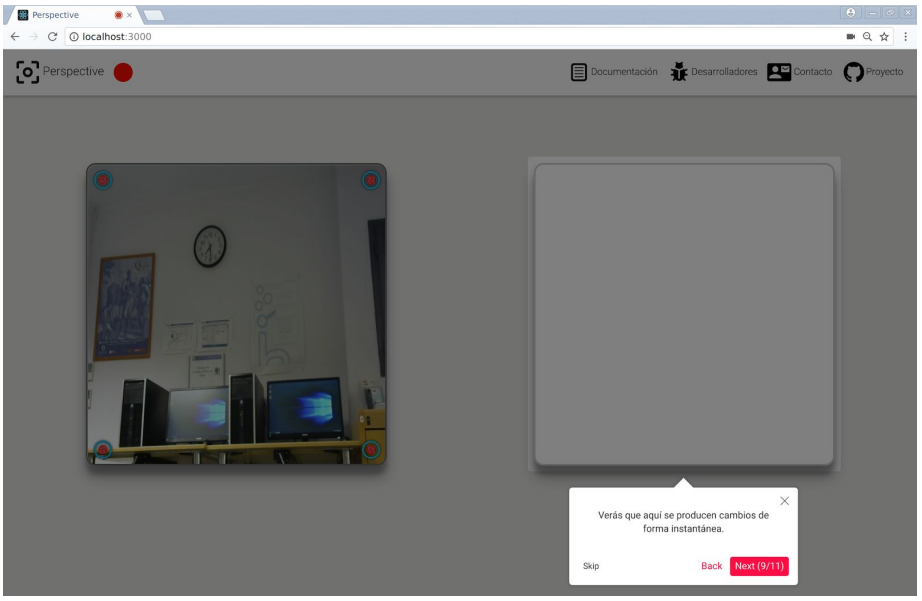
Ahora, en el cuadrado de la derecha observamos la imagen recuadrada transformada. Este procesamiento se realiza en tiempo real, de tal modo que si la webcam cambia de posición, los efectos serán distintos. El resultado de la transformación de perspectiva resulta interesante, pues la aplicación web simula que la webcam está en frente del cuadro, y no en un lateral, como sucede realmente.

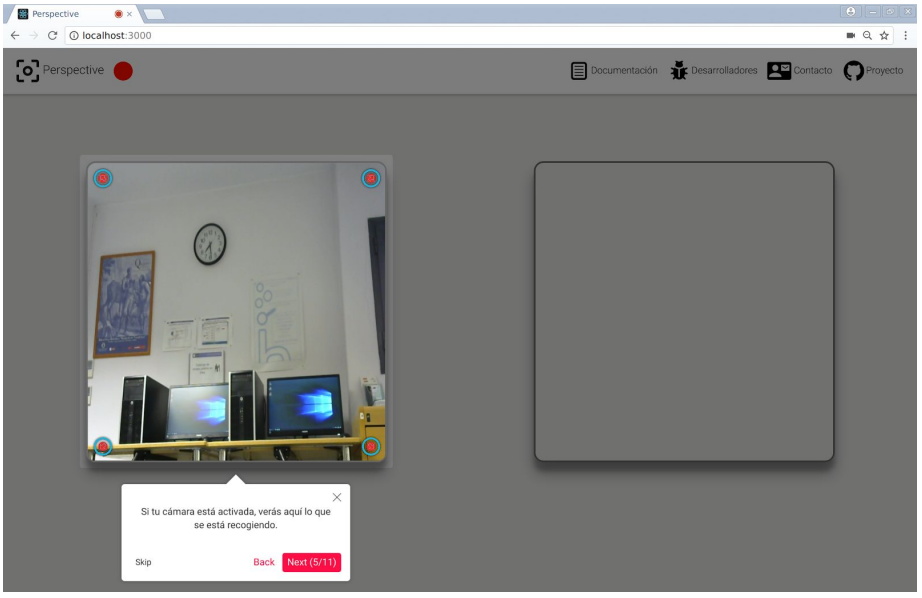
El usuario tiene a su disposición un tutorial interactivo, el cual se accede pulsando sobre el círculo rojo que se localiza entre los dos cuadrados principales.

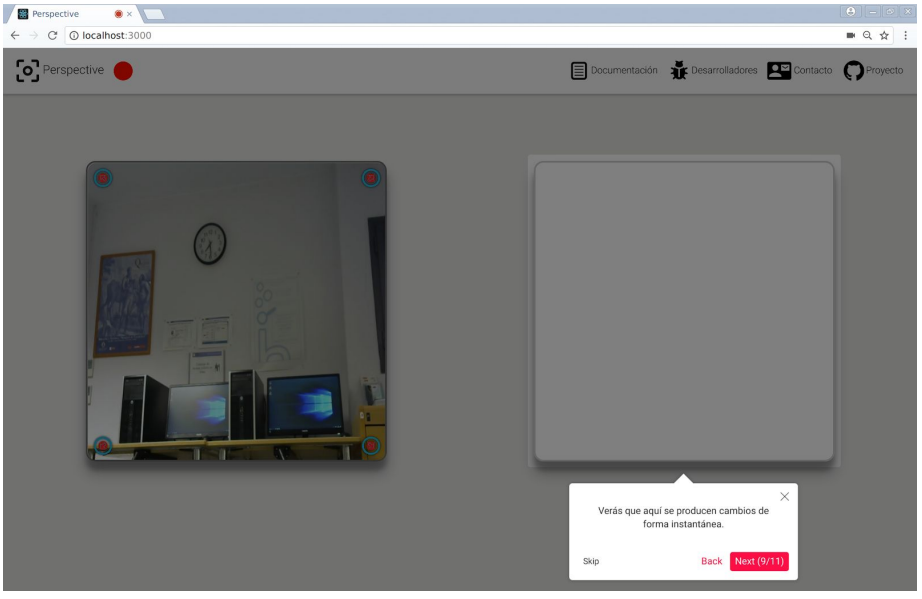


Al ser pulsado, nos aparecen diferentes mensajes colocados en ubicaciones donde se espera la interacción con el usuario.



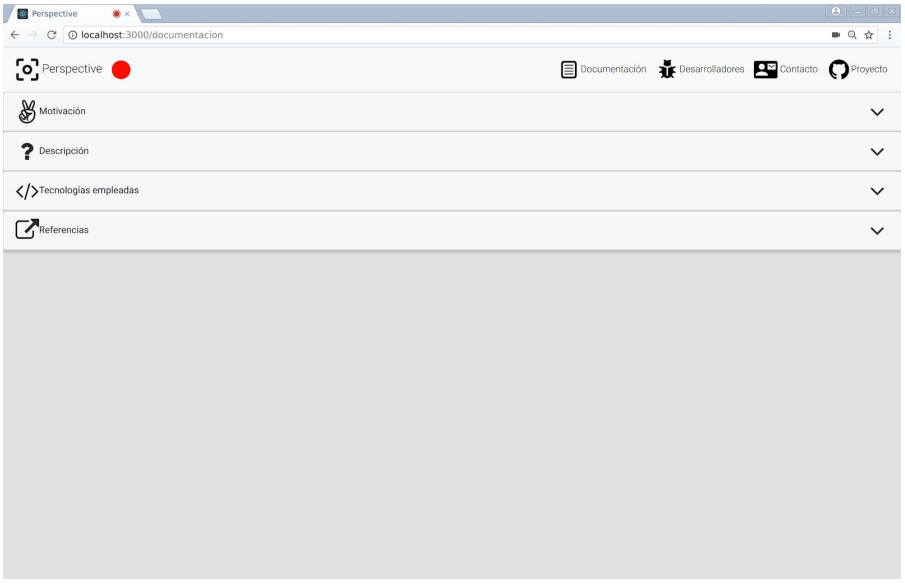

Perspective
localhost:3000
Perspective
Documentación
Desarrolladores
Contacto
Proyecto
Verás que aquí se producen cambios de forma instantánea.
Skip
Back
Next (9/11)


Perspective
localhost:3000
Perspective
Documentación
Desarrolladores
Contacto
Proyecto
Si tu cámara está activada, verás aquí lo que se está recogiendo.
Skip
Back
Next (5/11)

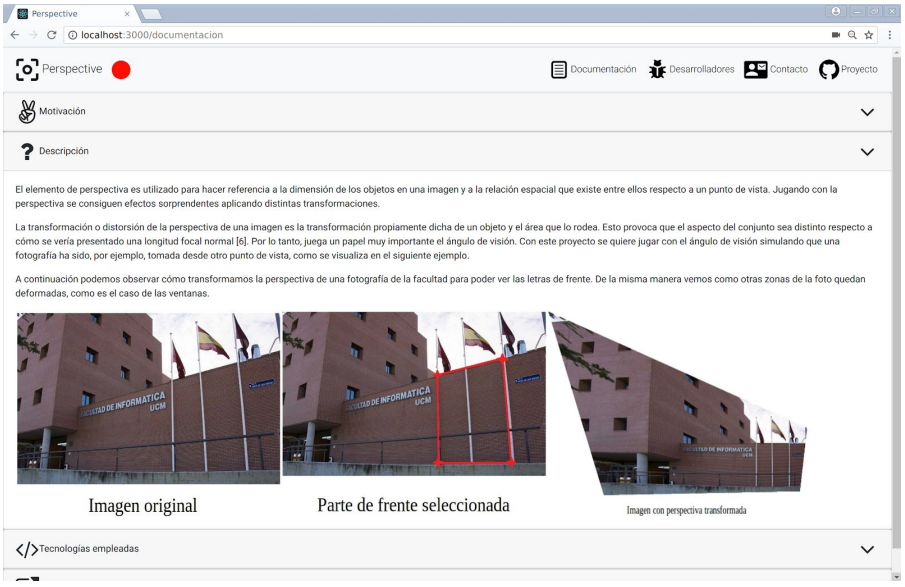

Perspective
localhost:3000
Perspective
Documentación
Desarrolladores
Contacto
Proyecto
Verás que aquí se producen cambios de forma instantánea.
Skip
Back
Next (9/11)

El tutorial está formado por 11 pasos cortos y sencillos de realizar. El mensaje se coloca en el lugar donde se espera la interacción del usuario.

En la barra superior disponemos de un enlace al proyecto Git (icono de la derecha del todo), y diferente documentación acerca del proyecto (icono documentación).



Si accedemos a una de las cuatro categorías pulsando sobre el nombre, se nos abrirá un pequeño texto con una explicación.



5. Tecnologías - Librerías de Terceros

En el siguiente apartado se nombrarán y profundizarán las diferentes tecnologías utilizadas para la realización de este proyecto, acompañadas de sus respectivas librerías correspondientes.

5.1. ReactJS

ReactJS es una librería Javascript desarrollada por Facebook, especialmente enfocada al desarrollo de entornos interactivos para navegador.

La premisa principal de ReactJS, más allá de la mejor estructuración del código y las posibilidades de intercomunicación entre elementos, reside en la problemática que suscita la ineficiencia de las modificaciones sobre el DOM. En la ejecución de código Javascript nativo, si se hace algún tipo de modificación minúscula en el DOM, obligatoriamente se tiene que actualizar por completo, suponiendo un coste perceptible a nivel computacional y de experiencia de usuario.

Para poder evitar este efecto negativo de Javascript, ReactJS propone incorporar un DOM virtual compuesto de JSX. Los JSX son una especie de variables similares a etiquetas HTML, pero cuya sintaxis está plagada de aspectos Javascript siendo objetos de este último lenguaje. El DOM virtual a nivel estructural viene a ser similar al real, pero es más ligero, y si es modificado también se recompone entero, pero el efecto sobre el DOM real es diferente y su coste a nivel computacional es menor. En contraste con respecto al DOM virtual, del DOM real sólo es modificado el elemento HTML al elemento JSX que se ha modificado en el DOM virtual, manteniéndose los elementos HTML restantes sin refrescarse.

Otra funcionalidad de interés de ReactJS es que se pueden estructurar los proyectos por componentes, que son básicamente clases Javascript que permiten generar JSX con el formato `<NombreClase/>` o `<NombreClase></NombreClase>`. Todos deben poseer un método render, que

será el llamado para generar su JSX, y pueden contener otros componentes dentro de sí encadenando sus generaciones. Además, permite centralizar en un único punto del documento HTML, sobre el que se hace el renderizado final, el ámbito de la WebApp a desarrollar, por lo que permite una mayor limpieza y legibilidad del código.

Además, los componentes tienen otra característica interesante. Los componentes permiten establecer un sistema de intercomunicación jerárquico, distintivo en ReactJS. En concreto, dicho sistema de intercomunicación se fundamenta en el uso de las propiedades y de los estados asociadas a los componentes React. Las propiedades son valores constantes que son preconfigurados de una forma parecida a como se configuran los atributos HTML, pero se pueden definir libremente. Puede servir para pasar valores constantes desde la clase padre a la hija, o incluso métodos que alteren o recojan el valor de las variables de estado de la clase padre. Por ello, son interesantes como modo de intercomunicación entre componentes descendente. Por otro lado, los estados son variables modificables asociadas a la componente que pueden ser alterados tanto por la clase propietaria de forma directa como por las clases hijas de forma indirecta mediante funciones de la clase padre configuradas propiedades mediante.

Otro aspecto a tener en cuenta es que, a causa de la nutrida comunidad que rodea a la tecnología, ReactJS está repleta de recursos para poder facilitar la programación visual de las páginas haciendo uso de librerías y paquetes que habilitan al usuario a disponer de una amplia cantidad de clases React predefinidas para múltiples propósitos: implementaciones de Konva, integraciones de bootstrap o material design, guías tutoriales sencillas… lo que permite un desarrollo ágil y sencillo de páginas web.

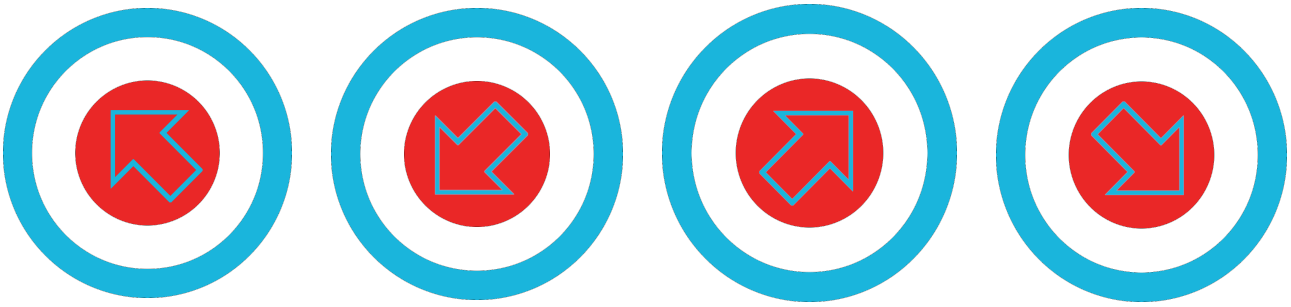
5.2. Konva

Konva es un framework basado en HTML5 Canvas JavaScript con el que se permite realizar animaciones en aplicaciones web. Permite dibujar objetos para que se visualicen en la pantalla y poder interactuar con ellos, ya que Konva da la posibilidad

de agregar a estos objetos listener de eventos. A partir de esta funcionalidad que nos proporciona el framework hemos implementado un sistema de puntos interactivos con el usuario para poder seleccionar el área de la imagen en la que el usuario desea realizar la transformación de la perspectiva.

Para empezar, hemos utilizado uno de los objetos que permite Konva, que s el objeto “Image”. Este objeto permite ser desplazado por un área. Además, permite recoger en cualquier momento las coordenadas de su ubicación, lo que posteriormente nos resulta de gran utilidad para las transformaciones de perspectiva.

Konva ofrece multitud de formas ya definidas, para nuestro proyecto hemos decidido utilizar una imagen como la que se ve a continuación.



Las flechas indican al usuario a qué esquina pertenece cada punto. Cara círculo es un objeto Image de Konva, y cada uno puede ser desplazado. Del mismo modo, cada círculo tiene sus coordenadas en la superficie donde se visualiza la imagen/vídeo. A partir de aquí, lo que realizamos es la recuperación de esas coordenadas para enviarselas a las funciones matemáticas encargadas de las transformaciones de la perspectiva.

Almacenar las coordenadas de los puntos también resulta interesante para poder pintar líneas entre los puntos de tal manera que el usuario visualice de manera más sencilla el área sobre el que se va a realizar las transformaciones matemáticas, como se puede observar en la imagen de acontinuación.

El uso de líneas, que resulta ser otro objeto de Konva, fue costoso. Como se explica en el apartado “Problemas enfrentados”, se requirió el uso de Threads en JavaScript a partir de una librería para poder mover a la par los puntos y las líneas.

En el momento de la integración de Konva con React se requirió el uso de React-Konva, un paquete disponible en React cuya forma es funcionar es similar a Konva para JavaScript con la salvedad que el código hay ajustarlo a la sintaxis de React.

6. Conceptos Matemáticos empleados

La imagen proveniente de la webcam va a ser tratada para que el área que se encuentra dentro de los cuatro puntos seleccionados se vea de frente.

Para conseguirlo, distorsionamos la imagen para que lo que aparece dentro de esa área se convierta en un cuadrado de 500 por 500 píxeles. Eso no es respetuoso con el aspect ratio original, pero nos permite hacernos a la idea de la forma que tendría esa área si la viéramos de frente.

Para conseguirlo, usamos la librería perspective-transform, cogiendo como puntos de destino los seleccionados por el usuario, y como destino los puntos (0,0), (0,500), (500,0) y (500,500), para transformar esa área en un cuadrado de 500x500 píxeles.

```
var srcCorners = [158, 64, 494, 69, 495, 404, 158, 404]; // Seleccionados por el usuario
var dstCorners = [0, 0, 0, 500, 500, 0, 500, 500];
var perspT = PerspT(srcCorners, dstCorners);
var t = perspT.coeffs;
t = [t[0], t[3], 0, t[6],
     t[1], t[4], 0, t[7],
     0   , 0   , 1, 0   ,
     t[2], t[5], 0, t[8]];
element.style.transform = "matrix3d(" + t.join(", ") + ")";
```

La librería nos genera la matriz homomórfica que podemos usar para transformar la imagen obtenida con la webcam con la propiedad transform y la función matrix3d() de CSS3.

7. Organización del trabajo

En cuanto a la organización del trabajo, éste se ha dividido en diversos módulos separados que han tratado de trabajar paralelamente en un principio. En concreto, dichos módulos fueron:

- Módulo de diseño visual de la aplicación
- Módulo de aspectos matemáticos de la aplicación
- Módulo de aspectos funcionales de la aplicación

El primer módulo fue asignado a Iván Aguilera y a Fernando Viñas, los cuales se centraron en el diseño de la página con la interactúa el usuario, tanto en los aspectos de colocación de los elementos, las decisiones de aspecto estético, así como el desarrollo del tutorial dinámico del sistema. Por tanto, se centralizó en la medida de lo posible aquello que implicase el uso de React.

El segundo módulo fue asignado a David Llop, encargado de conseguir trasladar los conocimientos de transformaciones matemáticas que ya teníamos desarrollados en el punto de partida del proyecto para la transformación de imágenes y elementos estáticos a su aplicación sobre elementos dinámicos. Este punto en principio es el más transversal, aunque pudo funcionar sin causar muchos problemas de forma independiente.

El tercer módulo fue asignado a Daniel García, encargado de implementar las esquinas que permitieran obtener las coordenadas de la fuente de vídeo original para su posterior recogida y transformación de lo recogido en el vídeo, así como los aspectos de feedback al usuario en torno al área que tales esquinas recoge. En este punto se centralizó todo lo relativo a Konva.

Sin embargo, debido a que en ciertos puntos han existido un gran número de interdependencias, como en el diseño de la interfaz y los aspectos funcionales de la aplicación, tal separación ha sufrido de resistencias que han lastrado un desarrollo

rápido del proyecto, como en el caso de integrar en React los aspectos funcionales de la aplicación desarrollados con Konva.

8. Problemas enfrentados

Durante su desarrollo, como en todo proyecto, nos hemos enfrentado a las más diversas asperezas en aras de conseguir nuestros objetivos. Entre ellas, podemos citar algunos problemas sufridos por nuestro grupo en los párrafos que siguen a continuación.

8.1. Bugs de refresco en algunos navegadores

Nos hemos percatado de que quizá al haber utilizado la tecnología React o al haber empleado la transformación de la imagen mediante la transformación 3D de CSS, algunos navegadores, bajo ciertas condiciones de inactividad de elementos visuales, dan problemas a la hora de refrescar los elementos del entorno visual. Concretamente, lo que sucede es que nuestro sistema congela la cuadrícula de emisión de vídeo con la perspectiva transformada. El modo en que React funciona nos da pie a sospechar en cuál puede ser la causa de que esto ocurra, pero también contamos con que ciertas incompatibilidades CSS sean las responsables.

Comencemos por nuestras sospechas con React. Debido a que React emplea un sistema de actualización del DOM real a través del DOM virtual, sólo actualizándose el primero cuando el DOM virtual sufre una transformación sustancial, y a la inactividad que este percibe al tener que refrescar la forma de la caja de vídeo sin más, creemos que el DOM virtual no notifica correctamente al DOM real cuándo debería actualizarse. Aquí es donde entran nuestras sospechas acerca de un posible conflicto con la matriz de transformación 3D de CSS, pues puede estorbar en esa retransformación. Pudimos comprobar que el efecto de la inactividad sucedía simplemente poniendo un console.log("...") en el código de CameraCapture, funcionando correctamente cuando el sistema mostraba en el inspector del navegador el mensaje logueado de forma indefinida. También pudimos percatarnos de que,

dependiendo del navegador usado, el bug se presenta o no, por lo que, si bien es una dificultad enfrentada y un problema conocido, no lo consideramos de nuestra de responsabilidad si no limitación del sistema.

8.2. Integración de la SideBar

En un comienzo habíamos planteado incorporar en nuestro entorno gráfico una barra lateral al estilo de las diseñadas mediante material design, que incorporara información periférica de nuestro proyecto: links sobre documentación, acceso al repositorio git del proyecto, acceso a la página git de sus autores...

Planteamos que el uso de Material Design Boostrap mediante un componente propio sería interesante, pero descubrimos que era de pago en algunas funcionalidades. En concreto, una de las que queríamos, las SideBar de MDB, estaba vetada en la versión gratuita y no teníamos la intención de pagar para poder acceder a ella. Debido a ello nos hemos dedicado a buscar una versión que fuese en primer lugar gratuita y en segundo lugar, a poder ser, libre.

Como alternativas planteamos emplear en primer lugar una SideBar encontrada por internet y cuyo código era de uso libre, pero se desechó rápidamente la idea debido a su complejidad. En segundo lugar, se planteó Material-UI, una librería React bastante interesante que nos permitía conseguir introducir en nuestro código componentes React que fueran directamente los elementos visuales que queríamos, y contaba en concreto con una SideBar diseñada bajo los parámetros de material design. Sin embargo, a causa de un problema de referencias más tozudo de lo normal planteamos desechar la idea inmediatamente.

En lugar de emplear la SideBar para las cuestiones que hemos citado anteriormente, hemos planteado incorporarlas en la NavBar de nuestro entorno. Quizá sea menos impresionante, pero hemos planteado que es mejor tender a lo más simple en aras de ser más efectivos.

8.3. Problemas con el sistema de guía al usuario

En primer lugar, habíamos planteado desarrollar un sistema de guía al usuario mediante avisos y feedback textual hecho por nosotros, mediante una caja de comentarios bajo los patrones estéticos de Material Design que nos permitiera emitir mensajes al usuario mientras interactúa con el sistema diseñado por nosotros a modo de tutorial. Habíamos dejar dicha caja de mensajes en la parte de abajo de la pantalla, e íbamos a emplear un array que nos permitiera modificar el mensaje en función de lo que quisiéramos mostrar.

Sin embargo, nos encontramos con que React, a la hora de generar un flujo de actividad controlado temporalmente mediante la función timeInterval(), no se lleva del todo bien con los eventos por defecto de Javascript. Si dejabamos su ejecución funcionando de un modo natural siempre mostraba el último mensaje setteado tras una secuencia correctamente correlada, aún poniendo mucho tiempo de intervalo. Esto ocurría incluso articulando correctamente la relación jerárquica de componentes para enlazar estados y propiedades entre ellos. Sospechamos que se debe a que ReactJS básicamente ignora estos callbacks, porque posiblemente tras la compilación según sus reglas tiene otro flujo de funcionamiento completamente distinto al que tiene Javascript.

Para superar este problema decidimos incorporar una nueva librería sugerida por uno de nuestros profesores, llamada Joydrive la cual nos permitió generar un tour guiado por la aplicación para poder explicar al usuario el funcionamiento de nuestro sistema de transformación de imagen. Su uso es muy sencillo, la integración con ReactJS es completamente natural y de hecho genera un acabado muy espectacular. En contrapartida, se pierde un aspecto del espíritu del primer acercamiento, que el sistema integrase el flujo guiado con el uso que hace de él el usuario. Por ejemplo, que una notificación le inste al usuario que mueva un punto del panel de recepción de vídeo de la cámara y espere a que el usuario efectivamente coja y arrastre algún punto. Joydrive permite dirigir la atención del usuario con marcadores rojos

parpadeantes, lo que es muy interesante, pero no permite incorporar una interacción entre sus tours y la actividad del usuario, que era lo que buscábamos en un primer momento. Sin embargo, el tour en sí nos parece igualmente muy útil.

8.4. Konva. Integración de Konva en React

El desarrollo del proyecto se dividió, como se ha comentado en apartados anteriores, en tres partes diferenciadas. Por un lado, estaba el desarrollo web utilizando la tecnología React. Por otro lado estaba el desarrollo de la parte matemática encargada de realizar las transformaciones necesarias para poder transformar la perspectiva de una imagen y por último estaba el desarrollo de la gestión de las coordenadas de los puntos selectores que el usuario utiliza para poder seleccionar el área de la imagen sobre la que se quiere realizar la transformación. Este último punto se realizó por separado de la parte del desarrollo web, de tal manera que el desarrollo web se realizó desde el principio utilizando React y el desarrollo de la gestión de los punto selectores se realizó en Javascript puro utilizando XAMPP para poder realizar las pruebas pertinentes. Tras estudiar los ejemplos que aparecen en la página de documentación de React, se consiguió dibujar cuatro puntos unidos por líneas. El primer problema fue conseguir que las líneas se movieran al mismo tiempo y hacia el mismo lugar que los puntos selectores. El problema residía en que el manejador de eventos que se ocupa del movimiento de los puntos ocupaba al navegador, por lo que al añadir otro manejador de eventos para poder mover las líneas se observaba que no se producía ningún efecto visual. Para solucionar este problema, se utilizó una librería Javascript que implementa Threads. Por lo tanto, la solución tomada fue crear un thread que se encargara de manejar el evento del movimiento de las líneas.

Una vez finalizado el desarrollo de la gestión de los puntos selectores se llegó a un nuevo problema, la integración de archivo Javascript que gestiona la selección de puntos con la web desarrollada con React. Ni Konva.js ni la librería ConcurentThread.js eran compatibles con React. Ante esta problemática se investigó la forma de solucionar este problema y se observó que React dispone de un

framework similar a Konva, llamado React-Konva, cuya funcionalidad es semejante a Konva, con la única salvedad de que React-Konva está adaptado al propio React.

9. Trabajo realizado

Del plan trazado originalmente, hemos conseguido alcanzar una serie de objetivos de los que nos hemos propuesto, detallados a continuación:

- Hemos desarrollado una página web como *front-end* de nuestra aplicación *de estilo minimalista*: un aspecto clave de las aplicaciones modernas es que, en primer lugar, tengan un diseño elegante, y en segundo lugar, no tengan la apariencia un farragoso panel de control de un avión. Es decir, que sea bello estéticamente y a la vez aúne aspectos que lo hagan usable. Nuestra web tiene el menor número de elementos posibles para hacer la interacción del usuario lo más sencilla posible, sólo moviendo los cuatro puntos en el panel central. El menú superior además es compacto, y no molesta en la interacción principal del usuario con la aplicación.
- Hemos conseguido *recoger el vídeo de la webcam* para su posterior transformación: un aspecto principal para nuestro proyecto en aras de poder transformar dicha fuente de vídeo. Este aspectos básicamente se ha realizado mediante las funcionalidades que nos ofrecen los navegadores modernos en cuanto a captación de la cámara como fuente de vídeo.
- Hemos conseguido *transformar la perspectiva de una imagen*, y lo que es más importante, hemos conseguido *transformar la perspectiva de un vídeo de forma dinámica*: es el aspecto nuclear de nuestro sistema. Mediante operaciones matriciales y las transformaciones con matrices 3D de CSS hemos podido conseguir la transformación de la perspectiva del vídeo que queríamos.
- Hemos conseguido *implementar un tutorial de uso*: nuestro sistema es capaz de guiar al usuario por la aplicación para que sepa cómo usarla en una serie de sencillos pasos. Ha sido implementada mediante Joyride, y si bien no permite interactuar con el entorno durante el tutorial, nos sigue siendo útil para mostrar al usuario la aplicación.

- Hemos *finalizado las diversas secciones de la página web*: que dan información periférica del proyecto mediante un selector de diversas páginas, como documentación, contactos del grupo desarrollador, acceso al repositorio Git, etc.

10. Trabajo no realizado

Por contra, hay otros aspectos que no hemos podido realizar. Fueron elementos planteados en el alcance inicial del proyecto como objetivos lejanos y opcionales, por lo que en el fondo no suponen problemas. En principio:

- No hemos realizado el *análisis del vídeo para la detección de puntos de color un específico evitando el uso de puntos selectores sobre la interfaz*: se pensó en primera instancia que sería interesante que el usuario no tuviera que seleccionar cuatro puntos en la propia imagen mediante el ratón, si no que pudiera establecer en el espacio real sobre el que quiere realizar la corrección de la perspectiva cuatro puntos con cuatro objetos reales de un color específico (por ejemplo, rojo) y que Perspective por sí misma, mediante el análisis de la fuente de vídeo capturada, pudiera detectar los cuatro puntos en el espacio real y además hemos pretendido que se realizara de forma automática la transformación del modo en que nosotros lo hemos hecho hasta el momento, pero por falta de tiempo no pudo desarrollarse.
- No hemos realizado la *transformación de la perspectiva de la imagen mediante captura de movimientos* con un giroscopio o algún artefacto similar: se pensó a su vez conseguir que el usuario pudiese conectar su teléfono móvil o un mando de la Wii al ordenador y que de alguna manera pudiese girar la perspectiva que había seleccionado mediante el movimiento capturado a través del dispositivo escogido, pero de nuevo, por falta de tiempo no pudo ni plantearse.

11. Conocimientos adquiridos. Conclusiones

Habiendo llegado al final del curso, y por lo tanto, a la finalización de un proyecto que ha durado tres meses podemos sacar en conjunto una serie de conclusiones.

En primer lugar, hemos aprendido una nueva forma de crear aplicaciones web a partir de React. Su inicialización no fue sencilla, puesto que requiere de un tiempo de adaptación. Para ello, se ha utilizado cursos encontrados en la web para aprender las nociones básicas de React.

En segundo lugar, hemos utilizado la herramienta GitHub de manera más profunda en relación con otros proyectos de otras asignaturas de la carrera/máster, donde su uso no era requerido. Hemos utilizado funcionalidades de GitHub que desconocíamos o que no utilizábamos, como las Issues para mostrar las tareas que se estaba desarrollando o los problemas a los que el proyecto se estaban enfrentado. También se ha hecho uso de las Wikis que proporciona GitHub para documentar el proyecto. Desde el documento de concepto del proyecto hasta la presente memoria, así como diferentes manuales de instalación o uso.

Por otro lado, hemos vivido como los primeros objetivos que se proponen nada más comenzar el proyecto no siempre es posible cumplirlos, ya sea por falta de tiempo o por falta de formación. Este aspecto está relacionado con la necesidad de tener una buena organización temporal de las tareas a realizar.

Por último, destacar que el desarrollo de este proyecto ha sido producido por el trabajo colaborativo de cuatro personas. Y como ocurre en mucho proyectos, esta tarea no resulta sencilla, puesto que hay cuadrar horarios, poner puestas en común, etc. De todos modos, como grupo no hemos tenido problemas reseñables, y el trabajo se ha realizado de manera fluida.

Si el tiempo volviese hacia atrás y el proyecto comenzara de nuevo, realizaríamos tareas de forma más conjunta, de tal manera, que si se debe realizar un curso de una tecnología de la cual va a estar apoyado todo el proyecto, realizaríamos todos el curso de manera conjunta y en el mismo periodo de tiempo para evitar así, problemas de incorporaciones.

En cuanto al uso de Git, procuraríamos hacer un uso más intensivo de la sección “Issues”, puesto que aunque han sido utilizadas para el proyecto, se podía haber usado más, pero al tratarse de algo nuevo, se requiere un tiempo de adaptación.

En cuanto a la organización de los miembros del grupo, no hemos tenido en los meses de duración del proyecto problemas de conflicto, por lo que esta organización se mantendría.

En cuanto a la definición de tareas, tendríamos más en cuenta en tiempo disponible para definir estas tareas y no proponer otras que no dé tiempo a realizar.

12. Bibliografía

- https://konvajs.github.io/
- https://github.com/lavrton/react-konva
- https://nodejs.org/es/
- https://material-ui.com/
- https://reactjs.org/
- https://www.npmjs.com/
- https://github.com/ReactTraining/react-router
- https://github.com/gilbarbara/react-joyride
- https://reactstrap.github.io/
- Curso de React
    - https://www.codecademy.com/
- Perspectiva para cámaras de tráfico
    - https://www.researchgate.net/figure/Figura-2-Ajuste-de-transformacion-de-perspectiva-y-obtencion-de-coordenadas-cartesianas-s_fig1_305755376
- Perspectiva para densidad de tráfico de cruce
    - http://utc.ices.cmu.edu/utc/tier-one-reports/Taylor_TSETFinalReport.pdf
- "Perspective Transformation." [Online]. Available:
    - https://bl.ocks.org/mbostock/10571478. [Accessed: 28-Feb-2018].
- P. N. Klein, Coding the Matrix: Linear Algebra through Applications to Computer Science, 1 edition. Newton, Mass.: Newtonian Press, 2013.
- "Perspective Transforms in JavaScript." [Online]. Available:
    - https://www.uncorkedstudios.com/blog/perspective-transforms-in-javascript/. [Accessed: 28-Feb-2018].
- https://www.researchgate.net/figure/Figura-2-Ajuste-de-transformacion-de-perspectiva-y-obtencion-de-coordenadas-cartesianas-s_fig1_305755376
- http://utc.ices.cmu.edu/utc/tier-one-reports/Taylor_TSETFinalReport.pdf
- https://es.wikipedia.org/wiki/Distorsi%C3%B3n_de_la_perspectiva