Trabajo de Fin de Grado

Control de asistencia basado en reconocimiento robusto de fiducials: aplicación para sistemas de sobremesa

Assistance control system based on robust recognition of fiducials: application for desktop

Sara Martín Molina

La Laguna, 05 de julio de 2015

D. **Leopoldo Acosta Sánchez**, con N.I.F. 42.165.911-B Catedrático de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como director

D. **Antonio Luis Morell González**, con N.I.F. 78.705.025-Z Personal Docente e Investigador en Formación adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como codirector

C E R T I F I C A (N)

Que la presente memoria titulada:

"Control de asistencia basado en reconocimiento robusto de fiducials: aplicación para sistemas de sobremesa"

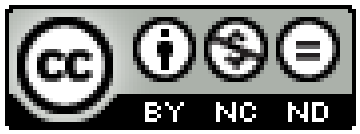
ha sido realizada bajo su dirección por D. **Sara Martín Molina**, con N.I.F. 54.108.519-F.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 5 de julio de 2015.

Agradecimientos

Me gustaría agradecer a mis tutores Leopoldo Acosta Sánchez y Antonio Lui Morell González por el apoyo y la ayuda prestada durante el desarrollo del trabajo y de esta memoria. También me gustaría agradecer a mi compañero y amigo Adrián González Martín por haberme ayudado en algunos aspectos del proyecto y la memoria y por último a mi familia, en concreto a mis padres, por haberme ayudado a llegar hasta aquí.

Licencia



Resumen

El objetivo de este trabajo ha sido el diseño y la implementación de una aplicación de escritorio para el control de la asistencia mediante el reconocimiento robusto de fiducials. La idea principal es que este sistema sea fiable y robusto y que nos permita ahorrar tiempo en el proceso de control de la asistencia. Para ello se ha diseñado un sistema que consta de dos partes, por un lado la aplicación de escritorio que haciendo uso de una webcam reconoce los fiducials y marca la asistencia del alumno en un hoja de cálculo de Google Drive y por otro lado una aplicación para dispositivos Android que genera un marcador a partir del NIU de cada alumno.

Palabras clave: *fiducial, control de asistencia.*

Abstract

The objective of this work has been the design and implementation of a desktop application to do control assistance using robust recognition of fiducials. The system have to be reliable, robust and easy enough to use in order to save time when a teacher wants to control the assistance to a lesson. With this objective in mind, we have designed a system which contains two parts. The first one is the desktop application. It uses a webcam to recognize the fiducials and then it marks the student's assistance on a spreadsheet Google Drive. The second one is an application for Android devices which generates a marker using the identification number of each student (NIU).

Keywords: *fiducial, control* assistance.

Índice General

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

En este trabajo se propone un sistema que permite a los profesores controlar la asistencia del alumnado de una forma rápida y robusta. Este sistema consta de diferentes elementos, por un lado un marcador personalizado que contiene el NIU codificado de cada alumno y por otro lado una aplicación de escritorio desarrollada en Java.

La aplicación de escritorio hace uso de una webcam para detectar los marcadores e internamente comprueba si ese marcador tiene codificado un NIU de los que están almacenados en la base de datos. Si es así muestra la información del alumno correspondiente y nos da la opción de pulsar un botón para confirmar la asistencia y que se marque en una hoja de cálculo de google drive.

Además de esta aplicación principal también se han desarrollado una serie de aplicaciones complementarias. Una de estas aplicaciones permite crear un marcador dado un NIU cualquiera. Y la otra permite meter a los alumnos en la base de datos.

Por otro lado, también se ha desarrollado una aplicación Android para los alumnos que les permitirá generar y visualizar su propio marcador en el móvil para no tener la necesidad de llevarlo impreso.

Capítulo 2. Antecedentes

Hoy en día existen ya muchos sistemas que permiten el control de la asistencia sobre todo en lugares de trabajo. Estos sistemas consisten normalmente en registrar la hora de entrada y de salida del empleado usando algún tipo de tarjeta que el empleado debe pasar por una determinada máquina.

En el ámbito escolar sin embargo es menos frecuente ver este tipo de sistemas, probablemente por el coste que podría acarrear. Por tanto lo que nos encontramos en colegios, institutos e incluso universidades son los sistemas tradicionales como pasar lista nombrando a cada alumno, o pasar el listado para que cada alumno firme y otros sistemas similares que no son del todo fiables y que además nos hacen perder tiempo.

Se han propuesto métodos más rápidos y fiables como el que desarrollaron dos alumnos de la Universidad Católica San Antonio de Murcia (UCAM) que funciona mediante el uso de tecnología de identificación por radiofrecuencias (RFID, Radio Frequency Identification). El sistema consta de tres elementos principales: una tarjeta de identificación que poseerá cada usuario; una antena situada en la entrada del aula y encargada de la lectura de las tarjetas; y un software encargado de procesar los datos adquiridos por la antena.

Este sistema se probó en algunas aulas de la UCAM y tras la prueba, los estudiantes y profesores que usaron el sistema de control de asistencia lo valoraron de forma muy positiva.

Sin duda esta propuesta es muy interesante pero tiene el inconveniente de que habría que hacer un gasto económico para ponerlo en marcha, ya que habría que instalar antenas en la entrada de todas las aulas y habría que crear tarjetas para todos los estudiantes.

Por eso en este trabajo se propone un sistema fiable, rápido y que se puede hacer con muy poca o ninguna inversión.

Capítulo 3.
Estado del arte

3.1 Tipos de marcadores

Los marcadores son útiles en muchas situaciones donde se necesita el reconocimiento de un objeto o determinar su pose con una alta fiabilidad. Las posibles aplicaciones de estos marcadores incluyen la realidad aumentada, etiquetas con mensajes que desencadenan un comportamiento, navegación de robots, y aplicaciones en las que necesitemos la posición relativa entre un objeto y la cámara.

3.1.1 Código de barras

Es un código basado en la representación mediante un conjunto de líneas paralelas verticales de distinto grosor y espaciado que es su conjunto contienen una determinada información. Las barras y los espacios representan pequeñas cadenas de caracteres. La correspondencia o mapeo entre la información y el código que la represente se denomina simbología.

Existen dos tipos de códigos de barras: los lineales y los bidimensionales.

Los códigos de barras lineales están implantados de forma global y se utilizan para identificar los artículos y poder realizar inventario de los mismos.

Figura 3.1 Código de barras lineal

Por otro lado, entre los distintos códigos de barras bidimensionales podemos destacar los códigos QR que son probablemente una de los códigos bidimensionales más utilizados hoy en día. Los códigos QR presentan 3 cuadrados en las esquinas que permiten detectar la posición del código respecto al lector.

Figura 3.2 Código de barras bidimensional (QR)

En estos códigos se pueden codificar URL, VCard, texto, E-mail, SMS, imágenes... De entre estos tipos de datos uno de los más comunes es el de codificar una URL ya que es una manera rápida y llamativa de promocionar la URL de un tienda, un producto...

3.1.2 AprilTag

Es un sistema de marcadores visual que se usa en diferentes ámbitos como la realidad aumentada, la robótica o la calibración de cámaras. Las etiquetas se pueden crear con una impresora estándar, y el software de detección calcula la posición, orientación y la identidad de las etiquetas en relación a la cámara.

El diseño y la codificación de las etiquetas se basan en un sistema de codificación lexicográfica casi óptima y el software de detección es robusto en distintas condiciones de iluminación y ángulos.

Está implementado tanto en Java como en C.

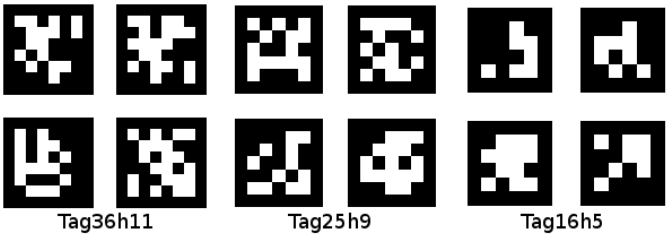


Figura 3.3 Marcadores de ejemplo de la librería AprilTag

3.1.3 ARToolKit

Es una librería de realidad aumentada mediante patrones. Permite posicionar y orientar la cámara con respecto a los marcadores y posicionar objetos 3D sobre los mismos. El lenguaje que se ha de usar con esta librería es C, aunque también permite otros lenguajes como Java o Matlab. Su licencia es GNU/GPL.

Figura 3.4 Marcadores de ejemplo de la librería ARToolkit

3.1.4 ArUco

ArUco es una librería para la realización de aplicaciones de realidad aumentada basada en OpenCV y cuyas principales funcionalidades son la creación y detección de marcadores y tableros de marcadores. Cuenta con la posibilidad de utilizar hasta 1024 marcadores distintos. El lenguaje de esta librería es C++ pero existe una versión en java llamada JArUco que es la que se ha usado en este trabajo.

De entre las distintas librerías se decidió usar ArUco para este trabajo por su robustez y sencillez de uso. ArUco permite la detección de los marcadores estando estos en cualquier posición cosa que no permite por ejemplo los códigos de barras lineales ya que para detectarlos deben estar alineados de una manera concreta respecto a la cámara para que sean detectados correctamente.

Además como punto a favor ArUco ha sido desarrollada por varios profesores de la universidad de Córdoba lo que ha sido útil a la hora de encontrar información en español sobre la librería.

Capítulo 4.
Objetivos

- Crear una aplicación que sea capaz de generar y detectar marcadores

El primer objetivo era conseguir una aplicación básica que fuera capaz de generar y detectar los marcadores de la librería ArUco.

- Encontrar una codificación adecuada para los marcadores

Uno de los objetivos más importantes era encontrar una codificación adecuada para poder relacionar cada alumno con un marcador único.

- Crear una aplicación móvil capaz de generar los marcadores

Para hacer más fácil y cómodo el uso de nuestro sistema se plantea como objetivo desarrollar una aplicación Android para que los alumnos puedan generar y llevar siempre encima sus marcadores.

- Conectar la aplicación a una base de datos y a una hoja de cálculo de Google Drive

Este objetivo se planteó teniendo en cuenta que todos los miembros de la ULL (Universidad de La Laguna) tienen acceso a una cuenta de Gmail y que hoy en día la forma más útil de almacenar la información para acceder a ella desde cualquier parte es el almacenamiento en la nube.

Capítulo 5. Fases y desarrollo del proyecto

5.1 Estudio previo

En este paso se exploraron las diferentes alternativas en cuanto al tipo de marcador que íbamos a usar. Para decidir el mejor marcador se valoraron aspectos como por ejemplo la robustez del sistema, la capacidad de detectar marcadores en diferentes posiciones e inclinaciones de la cámara o el marcador, su comportamiento en distintas condiciones de iluminación y la capacidad del marcador para codificar información.

Una vez que se encontró el marcador adecuado también se estudiaron las posibilidades en cuanto a los lenguajes de programación compatibles, sistemas operativos y herramientas complementarias.

5.2 Preparación del entorno

Una vez que se eligió ArUco como librería para la creación y detección de marcadores se debía preparar el entorno para poder utilizar dicha librería.

Como SO (sistema operativo) se decidió utilizar Ubuntu 14.04 ya que la aplicación está pensada para ser utilizada en las salas del centro de cálculo que mayoritariamente usan este SO.

En cuanto al lenguaje de programación para el desarrollo de la aplicación, teniendo en cuenta que existe una versión de ArUco para Java se decidió usar este lenguaje para entre otras cosas contribuir a que la aplicación sea portable a otros sistemas. A raíz de esto se decidió usar el IDE (Entorno integrado de Desarrollo, en inglés, Integrated Development Enviroment) Eclipse para el desarrollo de la aplicación.

A parte de lo mencionado anteriormente hay una serie de herramientas necesarias para el funcionamiento de ArUco y de la aplicación en general:

- OpenCV: es una librería de visión artificial multiplataforma que contiene más de 500 funciones como el reconocimiento de objetos, calibración de cámaras, visión estérea y visión robótica. Esta librería era el único requisito necesario para poder usar ArUco.

- H2: es un sistema administrador de bases de datos relacionales programado en Java que puede ser incorporado en aplicaciones Java o ejecutarse en modo cliente-servidor.

- Google Sheets API: es la API de Google para leer y modificar hojas de cálculo de Google Drive.

5.3 Creación de marcadores

Una vez configurado el entorno de trabajo el siguiente paso fue probar los ejemplos básicos que incluye la librería ArUco. El primer ejemplo era el de la creación de marcadores al que se le pasa por parámetro el identificador del marcador y el tamaño y nos genera el marcador en formato jpg.

```
1  public static void main(String[] args) throws Exception {
2      System.loadLibrary(Core.NATIVE_LIBRARY_NAME);
3      try{
4              if(args.length!=3){
5                  System.out.println("Usage: <makerid(0:1023)> outfile.↩
                       jpg sizeInPixels");
6                  System.exit(-1);
7              }
8
9              FiducidalMarkers fm = new FiducidalMarkers();
10             Mat marker = fm.createMarkerImage(Integer.parseInt(args↩
                   [0]),Integer.parseInt(args[2]));
11             Highgui.imwrite(args[1],marker);
12
13     }catch(Exception e){
14         System.out.println("Exception: " + e.getMessage());
15     }
16 }
```

Figura 5.1 Código java para la creación de un marcador

Para crear un marcador se ha de utilizar la clase *FiducialMarkers*. En la línea 9 se hace una llamada al constructor para tener un objeto *FiducialMarkers*. Con este objeto ya se podrían crear tanto marcadores como tableros. En este caso como lo que queremos es generar un marcador usamos el método *createMarkerImage* (línea 10) que recibe como primer parámetro un entero con el identificador del marcador y como segundo parámetro un entero con el tamaño en pixeles del lado del marcador.

Después, solamente quedaría guardar la imagen en el disco duro. Para ello se usa la función *imwrite* que se encuentra en el paquete *Highgui* de OpenCV. Esta función recibe como primer parámetro un *String* con la ruta donde se desea guardar la imagen y como segundo parámetro la imagen en sí (línea 11).

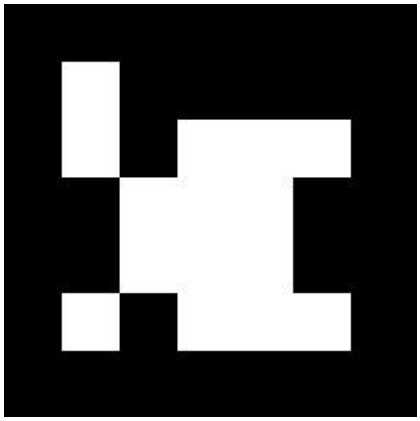


Figura 5.2 Marcador generado con la librería ArUco

5.4 Detección de marcadores

El paso siguiente era probar la detección de los marcadores. Para ello usamos de nuevo uno de los ejemplos de la librería ArUco.

Para detectar un marcador se ha de utilizar la clase MarkerDetector por tanto lo primero que hay que hacer es una llamada a su constructor.

```
1  MarkerDetector mDetector = new MarkerDetector();
```

Figura 5.3 Parte del código para la detección de un marcador

Una vez tenemos el objeto mDetector ya podremos detectar un marcador usando la función detect. A esta función se le pueden pasar distintos parámetros pero en este ejemplo solo se le pasa la imagen que contiene los marcadores y un ArrayList de objetos Marker donde se almacenaran los marcadores detectados.

```
1  ArrayList <Marker> Markers;
2  Markers = new ArrayList<>();
3  Mat InImage = Highgui.imread(args[0]);
4  mDetector.detect(InImage,Markers);
```

Figura 5.4 Parte del código para la detección de un marcador

Aunque el proceso para detectar los marcadores pueda parecer simple al necesitar solo unas líneas de código, el proceso que hace internamente la librería es más complejo. A continuación se describen los pasos que se siguen en este proceso de detección:

- Aplicar umbral adaptativo para obtener los bordes. (1)
- Encontrar los contornos. Después de esto, no solo los marcadores son detectados sino también muchos bordes no deseados.
    - Elimina bordes con un número pequeño de puntos. (2)
    - Se hace una aproximación poligonal de los contornos y se mantienen los contornos cóncavos con exactamente 4 esquinas. (3)
- Se ordenan las esquinas en dirección contraria a las agujas del reloj.
- Se eliminan también los rectángulos cerrados. Esto es necesario porque el umbral adaptativo normalmente detecta las partes internas y externas de los bordes del marcador. Nos quedamos con el borde más externo. (4)
- Identificación del marcador.
    - Se elimina la proyección perspectiva para obtener una visión frontal de la zona del rectángulo. (5)
    - Se aplica el algoritmo Otsu.
    - Identificación del código interno. El marcador está dividido en una cuadricula de 6x6 de las cuales las 5x5 celdas internas son las que

contienen la información y el resto son el borde externo negro. Primero se comprueba que existe el borde negro después se leen las celdas internas y se comprueba si proporcionan un código válido.

- Para los marcadores válidos se refinan las esquinas usando interpolación subpixel.
- Por último, si los parámetros de la cámara son proporcionados, se calculan los parámetros del marcador (rotación, posición en el espacio) con respecto a la cámara.

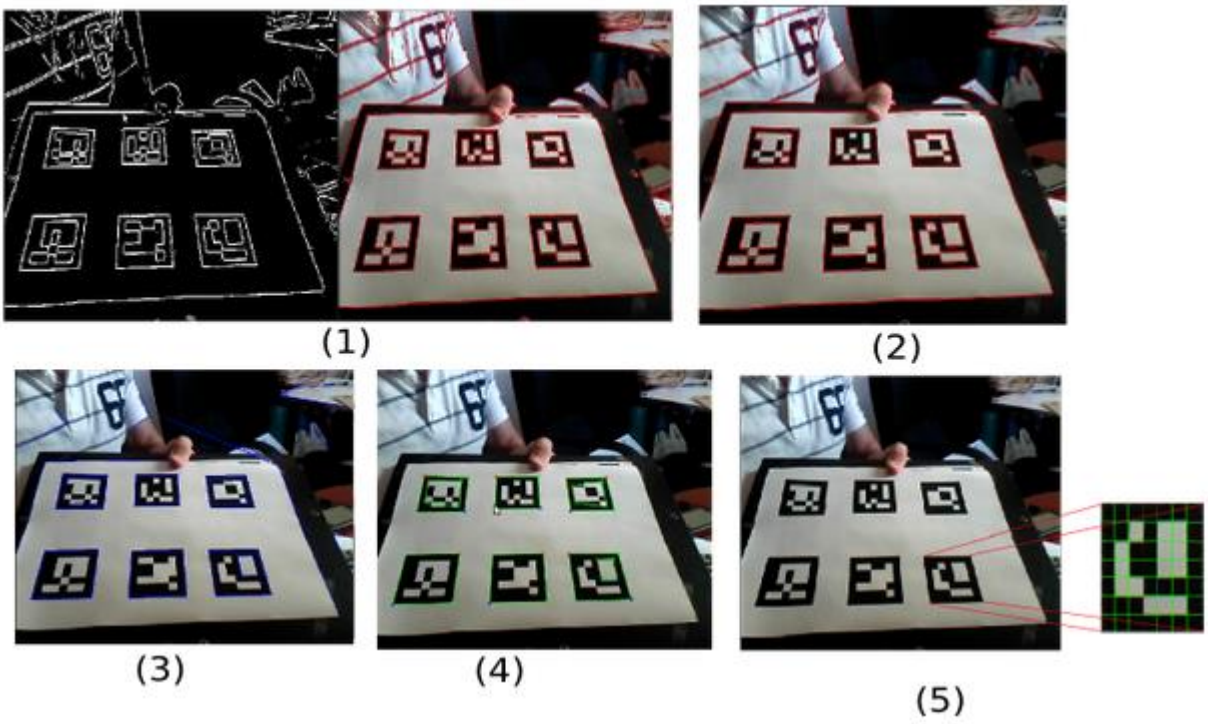


Figura 5.5 Proceso de imagen en la detección de marcadores [1]

5.5 Alternativas para la codificación de los marcadores

A la hora de crear los marcadores había que pensar la manera de asociar los marcadores a los alumnos. Para conseguir esto surgieron diferentes ideas:

Idea: usar el diccionario que proporciona ArUco y asociar cada marcador a un alumno.

Ventajas: no se necesitan crear marcadores porque ya los proporciona ArUco.

Desventajas: la limitación de marcadores que hacía que en algún momento tuviéramos que reutilizar los marcadores de alumnos que hubieran acabado la carrera para asignarlos a nuevos alumnos matriculados.

Idea: ampliar el diccionario de ArUco.

Ventajas: esta opción nos permite abarcar un mayor número de alumnos.

Desventajas: a largo plazo nos pasaría como la opción anterior y tendríamos que reutilizar los marcadores. Además para ampliar el diccionario había que modificar el código fuente de la librería de ArUco y era demasiado complicado para el tiempo que teníamos. Aun así se consiguió modificar la librería para generar hasta 4096 marcadores pero no se consiguió modificar el código para reconocer esos nuevos marcadores.

Idea: Utilizar el NIU de los alumnos para generar el marcador. Esta idea consiste en dividir el NIU en cuatro partes y codificar cada parte como un marcador individual.

Ventajas: tenemos marcadores "infinitos" ya que cada uno es generado a partir del NIU.

Desventajas: al dividir el NIU en cuatro partes se dejaban fuera los dos primeros dígitos del NIU. Esta opción se planteó puesto que ahora mismo los dos primeros dígitos son comunes a todos los alumnos. Aun así esta idea se descartó porque si algún día esos dígitos cambian ya no nos valdría esta codificación.

Idea: ampliar la idea anterior codificando los dos dígitos iniciales que dejábamos fuera usando las rotaciones internas de los marcadores.

Ventajas: tenemos las mismas ventajas que en el caso anterior y además codificamos los 10 dígitos del NIU.

Desventajas: al detectar un marcador este nos devuelve un número entero indicando el número de rotaciones de 90^{o} de dicho marcador con respecto a su

posición inicial. El problema de esto es que desde que el marcador esté rotado entre 1º y 89º ya devuelve un 1 indicando que el marcador está rotado 90º por lo que para poder usar estos datos para codificar parte del NIU se necesitaba además de los cuatro marcadores, un elemento de referencia para detectar la orientación inicial.

Idea: Utilizar el NIU de los alumnos para generar el marcador. Esta idea consiste en dividir el NIU en cinco partes y codificar cada parte como un marcador individual.

Ventajas: Esta alternativa sigue el planteamiento de las dos ideas anteriores pero usando un marcador más lo que nos permite codificar la totalidad de los dígitos del NIU.

Desventajas: la única desventaja puede ser el apartado estético ya que al contar con 5 marcadores se sale un poco del estilo habitual de este tipo de tarjetas.

5.6 Codificación de marcadores elegida

La codificación elegida ha sido la de dividir el NIU en cinco partes y usar un marcador para codificar cada parte. El usar varios marcadores tenía el inconveniente de que al detectarlos todos a la vez no sabíamos en qué orden colocarlos para formar el NIU que representan. Para solucionar esto se optó por añadir un dígito a cada una de las partes del NIU, como podemos ver en la figura siguiente.

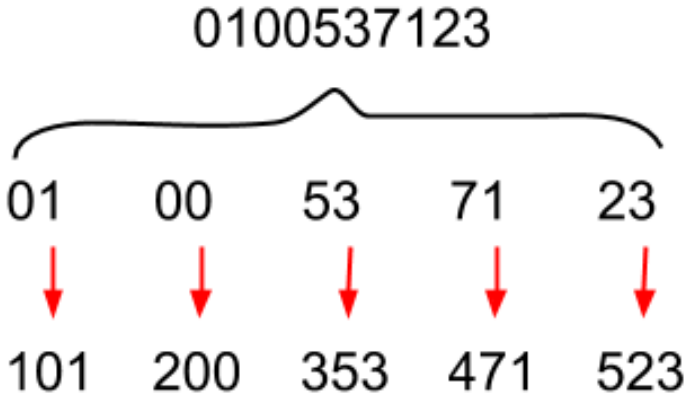


Figura 5.6 Codificación del NIU

De este modo sabemos que el marcador que este en el rango 100-199 representa los dos primeros dígitos del NIU, el que esté en el rango 200-299 representa el tercer y cuarto dígito del NIU y así sucesivamente. Por tanto la apariencia final del marcador sería la siguiente:

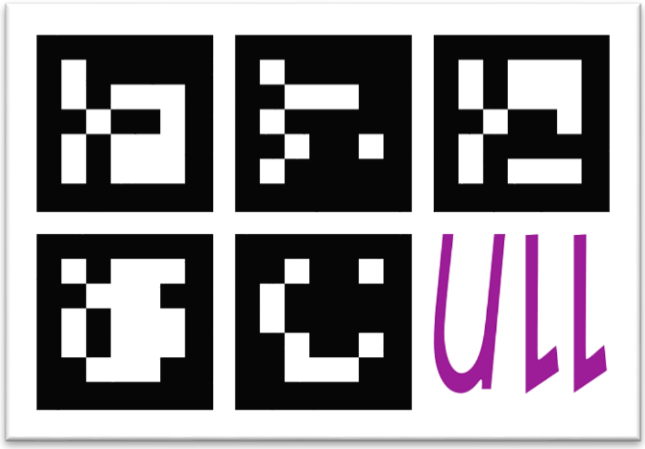


Figura 5.7 Marcador final

5.7 Conexión de la aplicación con Google

Para poder tener acceso a la hoja de cálculo de Google Drive en la que se va a marcar la asistencia de los alumnos, es necesario seguir una serie de pasos:

- El primer paso es registrar la aplicación en la consola de desarrolladores de Google para obtener el client_id y el secret_id de la aplicación.
- Después usando los parámetros que nos proporciona la consola de google se genera una URL que se abre en el navegador al iniciar la aplicación.
- Después de introducir nuestra cuenta de Gmail y aceptar los permisos necesarios se mostrara el código de autorización.
- Se copia y pega el código de autorización en la aplicación y esto genera el “access token” y el “refresh token”.
- Por último la aplicación utiliza el “access token” para hacer llamadas al API de Google.

El inconveniente de este sistema es que el “access token” caduca en una hora y por tanto la aplicación dejaría de funcionar porque no podría acceder a la API de Google.

La solución que existe es usar el “refresh token” para generar un nuevo “access token” y un nuevo “refresh token” asi solo se generaría el código de autorización una vez y después el “access token” se actualizaría automáticamente. Desafortunadamente esta opción no se ha conseguido implementar debido a la falta de información y de ejemplos sobre autorizaciones de Google en aplicaciones instaladas.

Lo que se ha hecho finalmente es guardar el “access token” en un fichero de texto y una vez que caduque hay que borrar el fichero y volver a realizar los pasos mencionados al principio de este apartado para generar un nuevo “access token”.

Por tanto, este es uno de los apartados más importantes que habría que mejorar en el futuro.

5.8 Base de datos y hoja de cálculo

Para rellenar la base de datos se creó un pequeño formulario en el que hay que introducir los datos del alumno incluyendo una foto para mejor la seguridad de la aplicación.

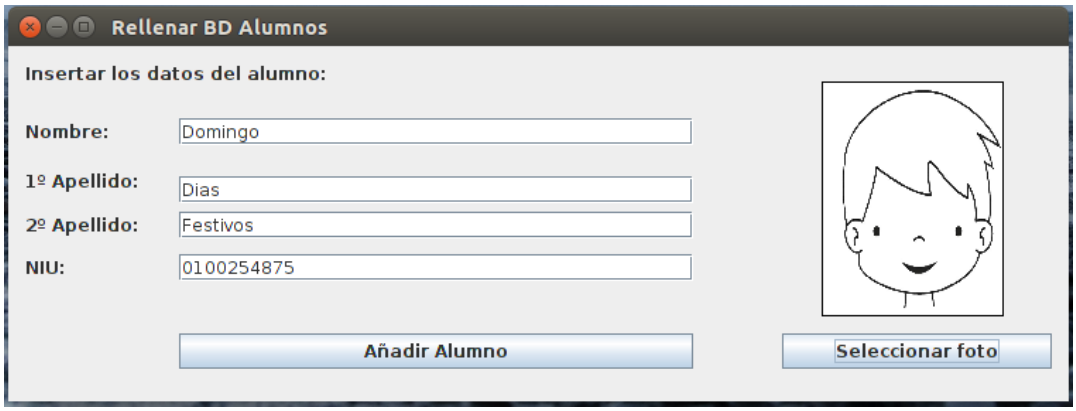


Figura 5.8 Aplicación para rellenar la base de datos

La estructura de la tabla de la base de datos es la siguiente:

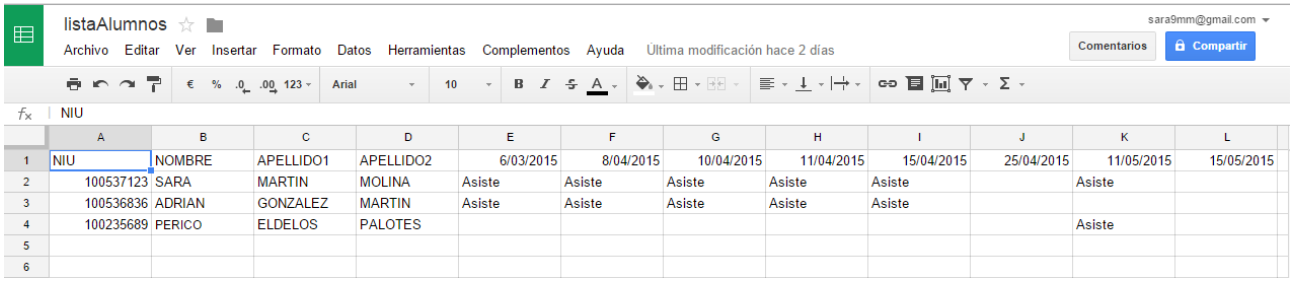
NIU	NOMBRE	APELLIDO1	APELLIDO2	FOTO
0100254875	Domingo	Dias	Festivos	/9j/4AAQSkZ.....

Tabla 5.1 Esquema de la base de datos

Todos los elementos de la tabla se guardan en la base de datos como cadenas de texto. En el caso de la foto lo que se ha hecho es codificarla en base64 para poder almacenarla también como una cadena.

Por otro lado, la aplicación se conecta a una hoja de cálculo de Google Drive, que deberá estar creada previamente, en la que se irá marcado la asistencia de los alumnos una vez que el profesor confirme su identidad.

Como vemos en la figura 5.9 tenemos la lista de los alumnos con su NIU y su nombre completo y a continuación podemos ver las fechas en las que se ha pasado lista y los alumnos que han asistido a clase.



	A	B	C	D	E	F	G	H	I	J	K	L
1	NIU	NOMBRE	APELLIDO1	APELLIDO2	6/03/2015	8/04/2015	10/04/2015	11/04/2015	15/04/2015	25/04/2015	11/05/2015	15/05/2015
2	100537123	SARA	MARTIN	MOLINA	Asiste	Asiste	Asiste	Asiste	Asiste		Asiste	
3	100536836	ADRIAN	GONZALEZ	MARTIN	Asiste	Asiste	Asiste	Asiste	Asiste			
4	100235689	PERICO	ELDELOS	PALOTES							Asiste	
5												
6												

Figura 5.9 Hoja de cálculo para marcar la asistencia

5.9 Aplicación móvil para crear marcadores

Para facilitar el que los alumnos lleven siempre su marcador encima sin tener que imprimirlo se ha desarrollado una sencilla aplicación Android que permite a los alumnos generar y visualizar su marcador.

5.9.1 Pantalla principal

En la siguiente figura se muestra una captura de la pantalla principal de la aplicación.

Figura 5.10 Pantalla principal en aplicación Android

Esta es la pantalla que veremos cuando no tengamos un marcador guardado en nuestro dispositivo. Ya que si la aplicación detecta que hay un marcador guardado lo abrirá directamente para agilizar el proceso de pasar lista. Si aun así queremos acceder a esta pantalla solo tenemos que pulsar el botón de "atrás" de nuestro dispositivo.

En el caso de que accedamos por primera vez a la aplicación tendremos la opción de generar o ver un marcador pulsando el botón correspondiente.

5.9.2 Generar marcador

Esta es la pantalla en la que se genera y se guarda el marcador. Sólo hay que introducir el NIU donde se indica y darle al botón "GENERAR". Después hay que presionar el botón "GUARDAR" para almacenar el marcador en nuestro dispositivo. Si se introduce un NIU incorrecto se mostrara un mensaje advirtiendo el error. También se muestra un mensaje si el marcador se ha guardado correctamente.

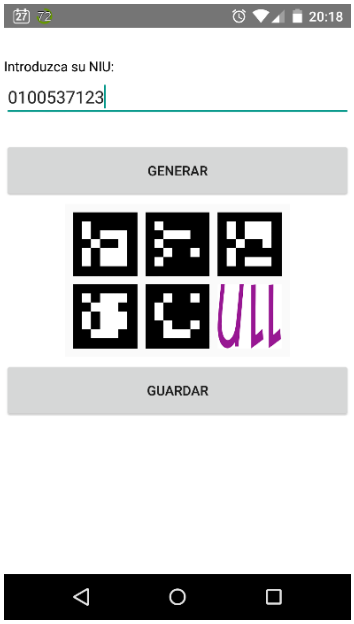


Figura 5.11 Pantalla de generar marcador en aplicación Android

5.9.3 Visualizar marcador

Por último podremos visualizar el marcador generada tanto de forma vertical como horizontal.

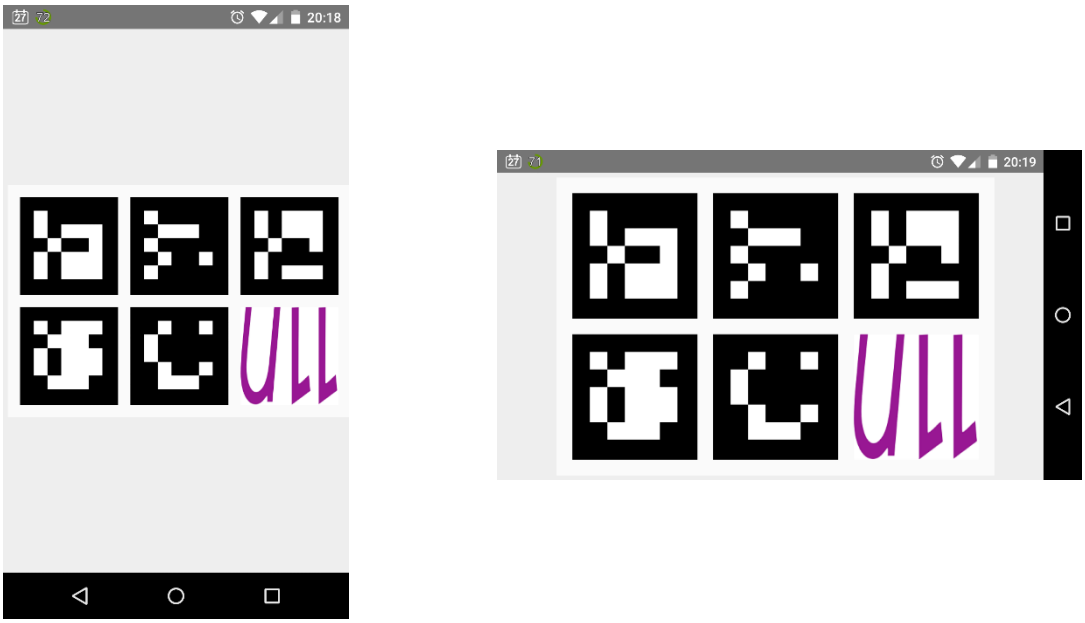


Figura 5.12 Pantalla de visualización del marcador en aplicación Android

5.10 Aplicación de escritorio

5.10.1 Interfaz

A la izquierda de la figura 5.13 se muestra la imagen que está captando la webcam en cada momento.

A la derecha podemos ver la información correspondiente al alumno asociado al marcador detectado. Se muestra el nombre completo, el NIU y la fecha actual, además de una foto para confirmar la identidad del alumno. Además el profesor podrá elegir en "Elegir la lista" la hoja de cálculo que desee usar para pasar lista. Cabe destacar que para que todo funcione, la hoja de cálculo tiene que tener la estructura vista en la figura 5.9.

Por último hay un botón que hay que pulsar cuando se haya comprobado la identidad del alumno y queramos escribir en la hoja de cálculo.

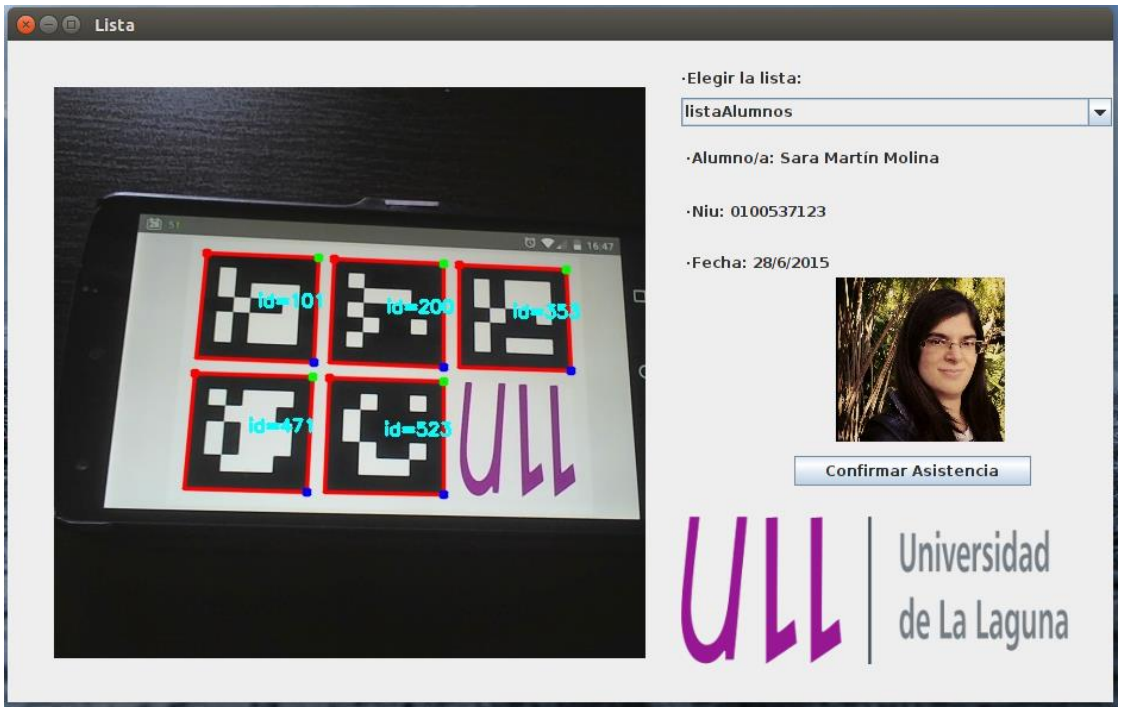


Figura 5.13 Interfaz de la aplicación de escritorio

5.10.2 Funcionamiento

El funcionamiento de la aplicación se divide en varios pasos:

1. Lo primero que hace la aplicación es abrir una página en el navegador para introducir nuestra cuenta de Gmail y así poder acceder a las hojas de cálculo de Google Drive.
2. Después la webcam empieza a capturar la imagen y cuando ponemos un marcador delante la aplicación decodifica ese marcador para obtener el NIU que está representando.
3. Una vez que tenemos el NIU lo buscamos en la base de datos. Si el NIU está en la base de datos mostramos la información del alumno.
4. Por último para registrar la asistencia del alumno hay que darle al botón de "Confirmar asistencia" y con esto se escribirá en la hoja de cálculo la palabra "asiste" en la fila correspondiente al alumno y la columna correspondiente a la fecha actual.

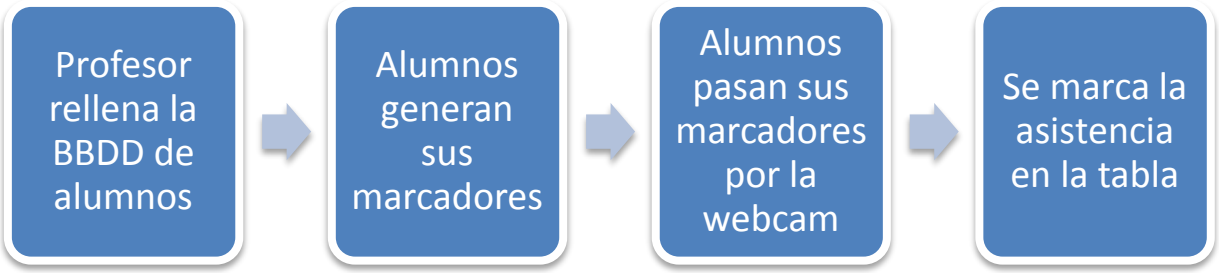


Figura 5.14 Diagrama de flujo del sistema

Capítulo 6. Conclusiones y líneas futuras

La idea de crear la aplicación descrita en esta memoria surgió para poder mejorar y acelerar el proceso del control de la asistencia. Después de haber estudiado diferentes alternativas y haber desarrollado dicha aplicación creo que esta es una opción a tener en cuenta en el futuro.

La aplicación desarrollada no conlleva ningún gasto económico ni para el profesor ni para el alumno puesto que el profesor solo necesita un ordenador con webcam y el alumno un dispositivo Android para generar y mostrar su marcador. Si no disponen de un dispositivo con Android también pueden llevarlo impreso. Además la librería ArUco funciona bastante bien y detecta los marcadores muy rápido y en distintas condiciones de iluminación.

Por otro lado, la aplicación también necesita algunas mejoras para hacerla totalmente funcional y fácil de usar. Lo ideal sería poder conectar dicha aplicación con alguna base de datos de la universidad para descargarnos la información de los alumnos ya que el sistema para rellenar la base de datos que se ha utilizado en este trabajo es demasiado lento al tener que meter los datos de los alumnos uno por uno. Con esta mejora se ahorraría mucho tiempo y tendríamos acceso a todos los listados de alumnos.

Otra mejora sería exportar la aplicación de escritorio a Android e incluso a otros sistemas operativos como IOS o Windows Phone. Esto permitirá no depender de si tenemos o no acceso a un ordenador con webcam ya que prácticamente la totalidad de los móviles disponen de cámara trasera. Además sería mucho más cómodo y se podría pasar lista en cualquier sitio ya que no necesitaríamos nada más que un Smartphone.

Por tanto, creo que la idea propuesta en este trabajo sería una gran alternativa a la típica hoja de firmas que se suele usar actualmente para el control de la asistencia pero como se menciona anteriormente todavía necesita algunas mejoras para hacer de esta aplicación una opción completa y viable.

Capítulo 7.
Summary and Conclusions

The idea of creating the application described in this project emerged because we want to improve and accelerate the process of control assistance. After the study of different alternatives and the development of this application I think that our system is going to be a good option in the future between the different control assistance systems.

The developed application does not generate any economic cost for the teacher or the student. The teacher only needs a computer with a webcam to use this system and the student needs an Android device to generate and display his marker. Each student can also print his marker if he/she doesn´t have an Android device. Moreover, the library ArUco works quite well because it detects very fast each marker in different lighting conditions.

In addition, we should add some functions in the future to make our application fully functional and easy to use. Ideally, the application should connect to any database of the university to download the students information. We have to introduce each student one by one now and this process is very slow if we have lots of students. With this improvement the teachers will save a lot of time and they will have access to all the student lists.

Another function that we should add in the future is the possibility of exporting the desktop application to Android and even to other operating systems such as iOS or Windows Phone. Using this Android application the teacher would not depend on whether or not he/she has access to a computer with a webcam because nowadays all phones have back camera. It will also be much more comfortable because we could use this system everywhere.

Therefore, I believe that the proposed idea in this project will be a great alternative to the signature sheet commonly used nowadays. The developed application is a good alternative to the control assistance systems but it still needs some improvements to make it a complete and viable option.

Capítulo 8. Presupuesto

Para calcular el presupuesto se ha tenido en cuenta el número de horas dedicadas a cada una de las fases de desarrollo de la aplicación.

Fases	Horas	Euros/Hora	Subtotal
Estudio previo	60	25	1500 €
Diseño	80	30	2400 €
Desarrollo	150	35	5240 €
Mejoras y solución de errores	40	35	1400 €
		Total	10540 €

Tabla 8.1 Presupuesto

Bibliografía

[1] ArUco: http://www.uco.es/investiga/grupos/ava/node/26

[2] JArUco: http://www.jArUco.hol.es/index.html

[3] H2 Database Engine: http://www.h2database.com/html/main.html

[4] H2 Wikipedia: http://es.wikipedia.org/wiki/H2_%28DBMS%29

[5] OpenCV: http://opencv.org/

[6] Google APIs: https://developers.google.com/identity/protocols/OAuth2

[7] Sheets API https://developers.google.com/google-apps/spreadsheets/

[8] S. Garrido-Jurado, R. Muñoz-Salinas, F.J. Madrid-Cuevas, M.J. Marín-Jiménez, Automatic generation and detection of highly reliable fiducial markers under occlusion, Pattern Recognition, Volume 47, Issue 6, June 2014, Pages 2280-2292, ISSN 0031-3203, http://dx.doi.org/10.1016/j.patcog.2014.01.005. (http://www.sciencedirect.com/science/article/pii/S0031320314000235)