Universidad de Buenos Aires
Facultad de Ciencias Exactas y Naturales
Departamento de Computación

Implementación de un Sistema de Identificación de Personas en Tiempo Real por Reconocimiento de Iris

Tesis presentada para optar al título de Licenciatura en Ciencias de la Computación de la Universidad de Buenos Aires

Alumno: Marcelo Luis Mottalli
<mmottalli@dc.uba.ar>

Directora de tesis: Dra. Marta Estela Mejail
Buenos Aires, Octubre de 2008

A mi mamá, que siempre me apoyó en todo lo que hice e hizo lo imposible por mí.
A Norberto, que probablemente sin él no huberia seguido esta carrera tan linda.
A Laura, mi compañera de aventuras.

Resumen

El reconocimiento de personas mediante el iris es aceptado como uno de los métodos biométricos más eficientes para la identificación, con el objetivo de controlar el acceso de individuos a edificios, oficinas, equipos y otros recursos protegidos.

Los métodos usuales de control de acceso involucran el recordar claves o códigos alfanuméricos, los cuales pueden ser olvidados fácilmente o, en el peor de los casos, robados. Es por esto que los saistemas biométricos basados en características morfológicas de la persona son cada vez más considerados como una solución para diferentes aplicaciones.

En particular, el reconocimiento mediante el iris presenta las ventajas de ser **no invasivo**, no requerir el **contacto físico** con ningún dispositivo y poseer una altísima **confiabilidad**.

El patrón del iris es **único** para cada individuo, **altamente diferenciable** entre individuos (baja cantidad de falsos positivos) y altamente **repetible** (baja cantidad de falsos negativos), a diferencia de otras características biométricas como por ejemplo el rostro.

El objetivo general de esta tesis es construir un sistema completo de identificación automática de personas basado en el reconocimiento del iris, probando algoritmos existentes y desarrollando algoritmos nuevos de procesamiento de imágenes, optimizados para el funcionamiento en tiempo real. El sistema estará compuesto por una cámara de video que se utilizará para capturar imágenes del ojo de las personas y el software necesario para procesar dichas imágenes. El sistema funcionará en **tiempo real**, con un mínimo de interacción entre el operador y el sistema.

Abstract

Recognition of persons by analysis of the iris texture is accepted as one of the most efficient biometric methods for identification, with the objective of controlling access of individuals to buildings, offices, equipment and other protected resources.

The usual control access methods involve memorizing passwords or alphanumeric codes, which can be easily forgotten or, in the worst case, stolen. This is why biometric systems based on morphologic features of persons are being increasingly considerated as a solution for different applications.

In particular, recognition of persons by iris has the advantages of being **non-invasive**, not requiring **physical contact** with any device of any kind and having enormous reliability.

The pattern of the iris is **unique** for every individual, it is **highly differentiable** between individuals (low amount of false positives) and **highly repeatable** (low amount of false negatives), unlike other biometric features such as the face.

The general objective of this thesis is building a complete and automatic system of identification of persons based on iris recognition, testing existing algorithms and developing new image processing algorithms optimised for real-time environments. The system will be composed of a video camera which will be used for capturing images of the eyes of the individuals, and the software needed to process such images. The system will work in **real time**, with minimum interaction between the operator and the system.

Agradecimientos

Gracias a Marta por haber tenido la idea y haberme dirigido en este trabajo que disfruté muchísimo haciendo.

Gracias a mis amigos de la facultad, que hicieron que la carrera, además de apasionante, fuera también divertida.

Gracias a Daniel y a Mariano por sus aportes y comentarios.

Gracias a Exactas y a la UBA, que me dieron la oportunidad de estudiar lo que más me gusta y, por si fuera poco, sin pedirme un peso a cambio.

Gracias a los docentes de la carrera, que siempre trataron a sus alumnos como futuros colegas y no como a estudiantes del montón.

¡Gracias a todos los que hicieron posible este trabajo!

Índice general

Capítulo 1

Introducción

La *biometría* (del griego *bios*, vida, y *metron*, medición) consiste en el estudio y aplicación de métodos matemáticos y estadísticos que tienen la finalidad de conocer la identidad de una persona. Estos métodos permiten analizar características físicas o del comportamiento de la persona que permiten distinguirla del resto de la población con cierto grado de certeza.

Algunas de las características físicas que permiten el reconocimiento de individuos son el rostro, las huellas dactilares, el ADN, la textura del iris y la geometría de la mano. Entre las características del comportamiento se pueden mencionar la firma, la voz e inclusive el patrón de tipeo de la persona en un teclado.

Para permitir el reconocimiento, una característica biométrica debe cumplir algunas condiciones:

- Debe ser **única** para cada individuo.
- Debe ser **capturable**, es decir, se debe poder capturar de alguna forma para poder ser analizada luego.
- Debe ser **repetible**, la característica se debe poder capturar las veces que sea necesario para la identificación.
- Debe ser **comparable** contra la misma característica pero obtenida de otro individuo
- Debe ser **estable a lo largo del tiempo**.
- Debe ser **difícil de falsificar o modificar**.

Algunas características deseables, pero no imprescindibles, son:

- Debe ser **fácil de obtener**. Por ejemplo, la huella dactilar es relativamente fácil de obtener comparada con el ADN, para lo cual se requieren técnicas avanzadas de extracción
- Debe ser **fácil de procesar**. Esto quiere decir que la característica biométrica debería permitir la identificación en el menor tiempo y con la menor complejidad posible.

Un *sistema biométrico* consiste en un conjuntos de técnicas que permiten capturar alguna característica biométrica de un individuo, almacenarla y luego utilizar esta característica almacenada para identificar o verificar la identidad del individuo cuando sea necesario. El ejemplo clásico de esto es el sistema de huellas dactilares utilizado por los gobiernos de todo el mundo. Mediante la utilización de la computadora, es posible implementar sistemas biométricos que funcionan de manera automática, capturando en forma digital la característica biométrica (por ejemplo, una fotografía digital del rostro o la firma escaneada de una persona) y guardando la misma utilizando alguna representación denominada *código* o *template* que luego permita una búsqueda ágil.

Hoy en día, la biometría es utilizada en muchas áreas fuera de la forense. Se utiliza por ejemplo como método de identificación en vez de utilizar contraseñas que pueden ser olvidadas o robadas, o como control de acceso a recursos restringidos.

1.1. El iris humano

El iris está compuesto de tejido fibroso llamado *estroma*, que conecta músculos encargados de contraer y dilatar la pupila en respuesta a los cambios de iluminación. El iris tiene un diámetro aproximado de 11 milímetros, y en su parte externa está rodeado de una capa blanca denominado *esclera* o *esclerótica*.

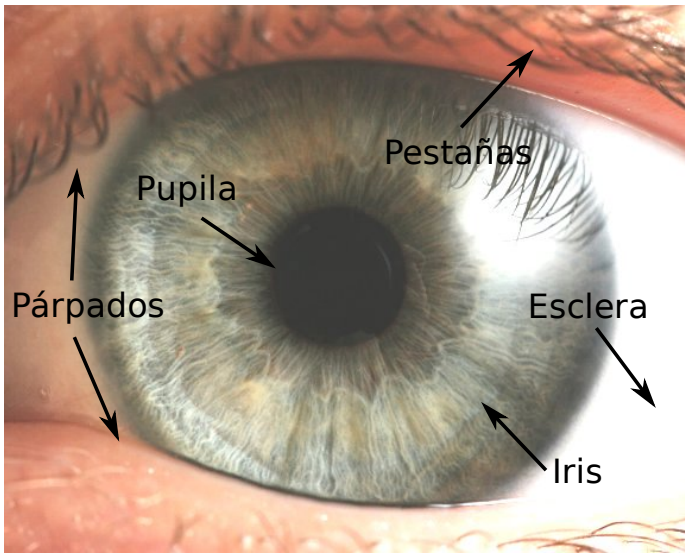


Figura 1.1: Diagrama del ojo humano

El tejido que compone el iris genera, como se puede ver en la figura 1.1, un patrón muy complejo, que es generado en forma completamente aleatoria durante los primeros meses de gestación y permanece constante a lo largo de toda la vida. Inclusive, el patrón del iris es independiente en cada ojo: no hay diferencia entre comparar la textura del iris del ojo derecho y del ojo izquierdo de una persona, con comparar la texura del ojo de dos personas diferentes.

Se estima que el iris es una de las características biométricas más ricas en cuanto a información que permite identificar un individuo: por ejemplo, mientras una huella digital tiene aproximadamente entre 40 y 60 características distinguibles, un iris tiene arriba de 240. Es esta complejidad y aleatoriedad de este patrón la que permite que el iris sea utilizado como característica biométrica.

Otras cualidades que permiten utilizar el iris como característica biométrica son:

- Es **externamente visible**: al ser parte del ojo humano, el iris es visible en condiciones normales, lo que permite que la textura sea capturada mediante una simple fotografía.
- Es sumamente **estable** a lo largo del tiempo.
- Es **imposible de modificar** por medios no-quirúrgicos. Inclusive algunos métodos quirúrgicos en el ojo como operaciones de cataratas mantienen constante la textura del iris [RM07].

1.2. Sistemas de reconocimiento de iris

Si bien la idea de utilizar el iris como característica biométrica tiene más de 100 años (propuesta por Bertillon en 1885), la idea de implementar un sistema de reconocimiento automático de iris surge en 1987 por Flom y Safir [FS87]. La idea fue patentada, pero no se conoce ninguna implementación.

La primera implementación de un sistema de reconocimiento de iris fue realizada por Daugman [Dau93]. Este trabajo definió las bases matemáticas y estadísticas que luego se utilizarían en casi todas las implementaciones siguientes. La mayoría de los sistemas de reconocimiento de iris disponibles comercialmente a la fecha utilizan los algoritmos desarrollados por Daugman. Se pueden mencionar también los sistemas desarrollados por Wildes et al. [WAG+94], Boles y Boashash [BB98], Lim et al. [LLBK01] y, más recientemente, el sistema desarrollado por Monro et al. [MRZ07].

Todos los sistemas de reconocimiento de iris funcionan capturando una imagen de ojo de la persona y, a partir de la textura del iris, generan un *código de iris* que luego es almacenado en una base de datos y utilizado para la identificación. Todo este proceso está dividido en varias etapas:

1. Primero, se hace una **captura** de una imagen del ojo de la persona
2. Una vez capturada la imagen, es necesario aislar la textura del iris y descartar el ruido producido por los párpados, las pestañas y cualquier reflejo ambiente. Esta etapa se denomina **segmentación**.
3. La textura del iris pasa por un proceso de **normalización** que permitirá invarianza frente a variaciones en el proceso de captura, como por ejemplo el tamaño del iris en la imagen o el diámetro de la pupila.
4. Esta textura normalizada es **codificada**, resultando una representación de la textura del iris que simplificará el proceso de identificación y verificación.
5. Una vez generado el código de iris, se pueden realizar dos acciones:
   - almacenar el código en una base de datos para referencia futura, o bien
   - realizar una búsqueda del código en una base de datos existente con el objetivo de **identificar** o **verificar la identidad** de la persona.

Estas etapas serán explicadas en detalle a lo largo del trabajo.

1.3. Objetivos del trabajo

El objetivo principal de este trabajo es implementar un sistema *completo* de reconocimiento de iris en tiempo real. Esto incluye distintos componentes, tanto de hardware como de software. Para esto será necesario implementar cada una de las etapas del proceso de reconocimiento, desde la captura de una imagen del iris hasta la identificación.

Se analizará también el estado del arte y se aportarán ideas con el objetivo de que el sistema funcione eficientemente y en la forma más automática posible.

1.3.1. Organización

El trabajo está organizado de la siguiente manera:

- En el capítulo 2 se analizarán los requerimientos que debe tener un sistema de **captura** que permita tomar imágenes del iris con suficiente calidad para la identificación. Se presentará también la solución de hardware implementada en este trabajo.

- En el capítulo 3 se analizará el problema de la **segmentación**, es decir, el problema de localizar correctamente el iris en la imagen. Se verán algunas soluciones existentes y finalmente se presentará un método optimizado de segmentación desarrollado para funcionar en un entorno en tiempo real.
- En el capítulo 4 se presentará un algoritmo destinado a **analizar la secuencia de video** obtenida del sistema de captura para conseguir una imagen del iris con la máxima calidad posible.
- En el capítulo 5 se hará un repaso de las distintas técnicas de **codificación** de la textura de iris, y de los métodos de **matching** asociados. Se describirá el método de codificación y matching implementado en el sistema.
- En el capítulo 6 se describirán en detalle los componentes del **sistema implementado**.
- En el capítulo 7 se analizarán los **resultados** de los algoritmos implementados para segmentación, codificación y matching. Se hará una descripción del marco teórico para medir los resultados y finalmente se presentarán los resultados en dicho marco.
- En el capítulo 8 se analizará el problema de la resistencia del sistema frente a posibles intentos de **falsificación**, y se presentará un método
- Finalmente, en el capítulo 9 se verán las **conclusiones** del trabajo, se analizarán los distintos **aportes** realizados y se presentarán ideas para **trabajo futuro**.

Además, el trabajo cuenta con un apéndice que detalla el funcionamiento de la librería de reconocimiento de iris implementada.

Capítulo 2

Sistema de captura

El proceso de reconocimiento de iris comienza por la captura de una imagen del iris de la persona a identificar. El componente encargado de este paso es el sistema de captura.

Un sistema de captura para reconocimiento de iris debe ser capaz de tomar imágenes del ojo del usuario con suficiente calidad como para poder analizar la textura del iris. Por esto, la performance final del sistema depende casi exclusivamente de la calidad de las imágenes obtenidas por el sistema de captura, ya que un sistema de captura de baja calidad no podrá capturar suficiente información de la textura del iris del usuario como para permitir un buen desempeño.

El propósito de este capítulo es analizar las distintas variables presentes en el sistema de captura, y cómo afectan estas variables al problema de capturar imágenes nítidas del iris. Finalmente, se describirá el sistema de captura implementado en este trabajo de acuerdo a los requerimientos expuestos.

Decimos que una imagen del iris es *nítida* cuando cumple dos condiciones:

- La zona del iris en la imagen se encuentra correctamente enfocada
- El iris posee una resolución que permita la identificación

El sistema de captura debe lidiar con dos problemas principales: poder capturar imágenes de alta calidad del iris y al mismo tiempo reducir al mínimo la complejidad de uso para el usuario. En cualquier sistema de captura, la calidad de las imágenes se ve afectada por tres conjuntos principales de parámetros: los parámetros de la cámara, los parámetros del lente y los parámetros de la iluminación.

- Los parámetros de la cámara son el tipo de sensor y la resolución de captura
- Los parámetros del lente son la longitud focal, el plano de enfoque, la apertura del diafragma y la profundidad de campo
- Los parámetros de iluminación son el tipo de iluminación y la dirección de iluminación

2.1. Parámetros de la cámara

La captura de imágenes nítidas del iris presenta varios problemas. Al tratarse de un objeto pequeño, de aproximadamente 11 milímetros de diámetro [Dau93], por lo general es necesario que la cámara se sitúe cerca del ojo para poder tomar una imagen lo suficientemente detallada.

La imagen capturada del ojo debe tener un tamaño en pixels suficientemente grande como para poder extraer la cantidad de información necesaria del iris. Es decir, no alcanza con que la

imagen del ojo sea nítida si tiene una resolución muy baja, ya que no va a ser posible extraer suficiente información de la textura del iris. Se estima que el radio del iris en una imagen no debe ser menor a 70 pixels para contener una cantidad mínima de información que permita el reconocimiento, mientras que comunmente se utilizan radios de entre 80 y 130 pixels [Dau04, WAG+94]. El estándar ISO para intercambio de información biométrica define un radio mínimo de 100 pixels para la captura [19705], o, equivalentemente, un diámetro de 200 pixels.

Para la captura de imágenes del ojo se pueden utilizar o bien una cámara de video o una cámara fotográfica. En el primer caso, se captura una secuencia de imágenes del ojo de la persona, y de la secuencia se extrae una o varias imágenes para la identificación. En el caso de la cámara fotográfica, se obtiene una única imagen, por lo general con mayor resolución que con una cámara de video. La solución más directa para un sistema de reconocimiento de iris en tiempo real es utilizar una cámara de video, mientras que un sistema de reconocimiento fuera de línea puede utilizar tanto una cámara de video como una cámara fotográfica.

Las cámaras cuentan con un sensor que permiten digitalizar las imágenes y su posterior procesamiento. En el mercado existen dos tipos de sensores: los CMOS (de las siglas en inglés, *Complementary Metal-Oxide Semiconductor*) y CCD (*Charge-Coupled Device*). Podríamos decir que, a fines prácticos, cualquiera de los dos tipos de sensores tiene un desempeño equivalente, por lo que esto no impactará en el funcionamiento del sistema.

Ya sea que se utilice una cámara de video o una cámara fotográfica, es necesario definir ciertos parámetros de la cámara para que sea posible capturar imágenes del iris con la máxima nitidez posible.

2.2. Parámetros del lente

Los parámetros del lente abarcan la longitud focal, la distancia de enfoque y la apertura del diafragma.

2.2.1. Longitud focal

La longitud focal del lente está relacionada con el ángulo de captura del mismo.Como regla general, mientras mayor sea la longitud focal del lente, mayor podrá ser la distancia a la que se deberá ubicar el ojo, ya que con una longitud focal grande es posible tener un ángulo de captura pequeño, y la imagen del iris tendrá un tamaño adecuado.

La longitud focal del lente dependerá de tres factores [Sou06]:

- El *ancho del objeto* a capturar, W_o (en este caso, el ojo)[1]
- El *ancho del sensor* W_s de la cámara
- La *distancia de trabajo*, D_t, que es la distancia a la cual estará posicionado el objeto (respecto al lente de la cámara)

La longitud focal f del lente que permitirá una distancia de trabajo D_t se podrá calcular a partir de estos parámetros utilizando la siguiente fórmula:

$$f = \frac{D_t \cdot W_s}{W_o + W_s} \tag{2.1}$$

En el caso particular de la captura de imágenes del ojo, se tiene que el tamaño del mismo es de aproximadamente 50mm de ancho. El tamaño del sensor es variable y depende del tipo de cámara que se utilice. Los tamaños más comunes para un sensor del tipo CCD pueden verse en el cuadro 2.1.

[1] No se considera el alto del ojo ya que es despreciable frente al ancho

Formato del sensor (pulgadas)	Ancho (mm)	Alto (mm)
1/4″	3,2	2,4
1/3″	4,8	3,6
1/2″	6,4	4,8
2/3″	8,8	6,6
1″	12,8	9,6

Cuadro 2.1: Tamaños comunes de sensores CCD

La distancia de trabajo puede ser variable e ir desde unos pocos centímetros hasta una distancia de algunos metros para algunas implementaciones de sistemas de detección de iris, siendo lo común una distancia de entre 30 y 60 centímetros.

Por ejemplo, para un sensor de un tercio de pulgada (1/3″, de 4,8 x 3,6 mm) y un tamaño de ojo de 50 mm, la longitud focal para una distancia de trabajo de 30 cm será:

$$f = \frac{300 \cdot 4{,}8}{50 + 4{,}8} \approx 26\text{mm} \tag{2.2}$$

En forma inversa, si por ejemplo se posee un lente de $f = 6$mm, se estará limitando la distancia de trabajo a:

$$D_t = \frac{f \cdot (W_o + W_s)}{W_s} = \frac{6 \cdot (50 + 4{,}8)}{4{,}8} = 68{,}5\text{mm} \tag{2.3}$$

es decir, poco más de 6 centímetros.

La mayoría de los lentes existentes hoy en día en el mercado son los denominados *lentes varifocales*, o también llamados *lentes zoom*, que permiten variar su distancia focal. Así, se pueden conseguir por ejemplo lentes cuya distancia focal varía entre los 5 a 25mm, 6 a 60mm, etc. Estos lentes son los más versátiles a la hora de implementar un sistema de reconocimiento de iris ya que permiten variar la distancia de trabajo sin la necesidad de cambiar el lente.

Cabe mencionar que mientras mayor sea la distancia de trabajo, mayor dificultad tendrá el usuario en posicionarse para que su ojo quede alineado con el lente de la cámara. Además, al ser pequeño el ángulo de captura, es susceptible a que el más mínimo desplazamiento de la cabeza ocasione que el ojo se salga del cuadro de la imagen, o también que produzca imágenes borrosas causadas por el movimiento de la cabeza (esto se denomina en inglés *motion blur*). Como contraparte, una distancia de trabajo muy pequeña puede causar incomodidad al usuario al tener que acercar demasiado su ojo al lente.

2.2.2. Distancia de enfoque

Si bien la distancia de trabajo define la distancia óptima a la cual debe posicionarse el ojo para que el mismo ocupe toda la imagen, esto no quiere decir que sea la distancia para la cual la imagen tendrá una nitidez máxima. La nitidez máxima de la imagen se consigue cuando el objeto está a una distancia del lente igual a la distancia de su plano de enfoque (que no debe confundirse con su *distancia focal*).

El plano de enfoque de un lente es un plano imaginario, perpendicular al eje del lente, en el cual todos los puntos pertenecientes a dicho plano están perfectamente enfocados. Este plano de enfoque se expresa como la distancia a la cual está el plano respecto del lente, por ejemplo, si un lente posee su plano de enfoque a 10cm, esto quiere decir que todos los puntos que estén en el plano a 10cm del lente serán perfectamente enfocados.

Cuando un punto no pertenece al plano de enfoque, se dice que el mismo está "desenfocado". Visualmente, esto se percibe como una pérdida de nitidez en el punto en cuestión. El punto

aparecerá en la imagen como un círculo, llamado "círculo de confusión". Mientras más alejado esté el punto del plano de enfoque, mayor será el diámetro del círculo de confusión.

La mayoría de los lentes permite variar su plano de enfoque. Existen dos tipos de lentes de acuerdo a cómo varían dicho plano: por un lado están los *lentes manuales* y por otro los *lentes autofocus*. Los lentes manuales poseen un aro en su cuerpo que permite ajustar la distancia del plano de enfoque, mientras que los lentes autofocus poseen un mecanismo por hardware que ajusta su plano de enfoque automáticamente a fin de conseguir la imagen más nítida posible.

La ventaja de los lentes autofocus frente a los manuales es que las imágenes capturadas siempre tienen una nitidez máxima. Sin embargo, esto puede no ser una ventaja para el reconocimiento de iris. Cuando se está tomando una imagen del ojo, es posible que el lente tenga problemas para enfocar automáticamente la región del iris, ya que puede enfocar en cambio a las pestañas, o a las cejas, que se encuentran en un plano distinto al iris y por lo tanto no se obtendrá una nitidez máxima.

Los lentes manuales poseen la ventaja de que son mucho más flexibles a la hora de establecer los parámetros del sistema. Se puede establecer su plano focal a una distancia igual a la distancia de trabajo para obtener imágenes de un tamaño y nitidez óptima.

Con un lente manual, sin embargo, se deberá contar con algún mecanismo que permita establecer que el iris del usuario está aproximadamente a la distancia adecuada (igual a la distancia del plano de enfoque). En este caso se puede contar con una lógica por software que permita, para una imagen obtenida por la cámara, establecer la nitidez del área del iris.

Una tercera opción sería contar con un lente cuyo plano de enfoque pueda ser ajustada en tiempo real mediante programación, lo que permitiría ajustar, sin intervención del usuario, el plano de enfoque adecuado para obtener la mejor nitidez en el área del iris. Sin embargo, estos lentes son difíciles de conseguir y resultan bastante costosos.

Una limitación de cualquier tipo de lente puede ser su distancia mínima de enfoque, que puede ser mayor a la distancia de trabajo. Esto se puede solucionar usando tubos de extensión, que alejan el lente del sensor de la cámara disminuyendo su distancia mínima de enfoque.

2.2.3. Apertura de diafragma y profundidad de campo

Además del plano de enfoque, el lente puede variar también su *diafragma* (también denominado *iris*), que es un anillo que regula la cantidad de luz que llega al sensor[2].

La cantidad de luz que recibe el sensor de la cámara dependerá también de la longitud focal del lente, ya que un lente de mayor longitud focal tiene una pérdida de luz a lo largo de su longitud. Es por ello que no conviene representar la apertura del diafragma únicamente como su diámetro, sino como una función de su diametro d y la longitud focal f del lente:

$$F = \frac{f}{d} \tag{2.4}$$

Este número F es un número sin dimensión que representa la cantidad de luz que recibirá el sensor.

De la misma forma que con los lentes manuales y autofocus, existen lentes cuyo ajuste de diafragma puede ser manual o automático. Estos últimos son denominados *lentes autoiris*, y ajustan el diámetro del diafragma automáticamente de acuerdo a la intensidad de la luz recibida por el lente.

El diámetro del diafragma varía en forma inversa a la cantidad de luz disponible: si hay poca luz, el diafragma deberá ser grande a fin de capturar mayor cantidad de luz, mientras que si la cantidad de luz es mucha, el diámetro del diafragma deberá ser pequeño para poder bloquear

[2]En realidad, el diafragma y el iris son dos cosas distintas. El diafragma es el agujero por donde pasa la luz hacia el sensor, mientras que el iris es el mecanismo que permite variar el diámetro de este agujero. Es análogo a la pupila y el iris humano, respectivamente.

Formato del sensor (pulgadas)	δ (mm)
1/4″	0,008
1/3″	0,011
1/2″	0,016
2/3″	0,021
1″	0,03

Cuadro 2.2: Valores para el diámetro máximo δ del círculo de confusión de acuerdo al formato del sensor (extraído de www.rainbowcctv.com/tech/depth.html)

parte de esa luz y no provocar una saturación del sensor de la cámara (lo que dará como resultado una imagen blanca).

Además de regular la cantidad de luz que recibe la cámara, el diafragma permite variar la *profundidad de campo* del sistema. La profundidad de campo representa el rango en el cual un punto del espacio proyecta un círculo de confusión lo suficientemente pequeño como para ser considerado en foco.

Para poder calcular la profundidad de campo del sistema, es necesario conocer dos valores: la *distancia cercana de enfoque*, ΔL_c, y la *distancia lejana de enfoque*, ΔL_l. Estas distancias marcan los límites de la región del espacio que se considera "enfocada".

Para calcular estos valores, se introduce una variable δ que representa el tamaño máximo del círculo de confusión de un punto del espacio en el sensor de la cámara para que el mismo aparezca enfocado en la imagen. Dicho parámetro depende del formato del sensor (ver el cuadro 2.2).

Definido el valor de δ, las fórmulas para ΔL_c y ΔL_l son entonces:

$$\begin{aligned} \Delta L_c &= \frac{\delta \cdot FD_t^2}{f^2 + \delta F D_t} \\ \Delta L_l &= \frac{\delta \cdot FD_t^2}{f^2 - \delta F D_t} \end{aligned} \tag{2.5}$$

La profundidad de campo total es la suma de estos dos valores [HCTW06]:

$$\text{Profundidad de campo} = \Delta L_c + \Delta L_l = \frac{2\delta \cdot F(fD_t)^2}{f^4 - (\delta \cdot FD_t)^2} \tag{2.6}$$

Los distintos parámetros que intervienen en la profundidad de campo están ilustrados en la figura 2.1.

La profundidad de campo es un parámetro muy importante en un sistema de captura de iris, ya que define el rango en el cual se deberá situar el ojo de la persona para poder ser enfocado correctamente. Una profundidad de campo de unos pocos centímetros o menos implicará que el usuario tenga que ubicar su ojo en una posición casi exacta, ya que una pequeña desviación ocasionará que la imagen del ojo no salga enfocada. Esto puede ser una molestia para el usuario.

Una profundidad de campo más grande permite al usuario un margen de error aceptable en el posicionamiento de su ojo respecto a la distancia de trabajo, sin degradar demasiado la performance del sistema.

La profundidad de campo también depende de la distancia de trabajo D_t, definida anteriormente. Una distancia pequeña entre la cámara y el iris ocasiona que la profundidad de campo sea muy reducida, y esto también puede resultar una molestia para el usuario.

Se puede ver entonces cómo la apertura del diafragma impacta directamente en la performance del sistema variando la profundidad de campo y el rango en el cual el usuario debe posicionarse. Luego, será necesario ajustar la apertura del diafragma en función de la profundidad

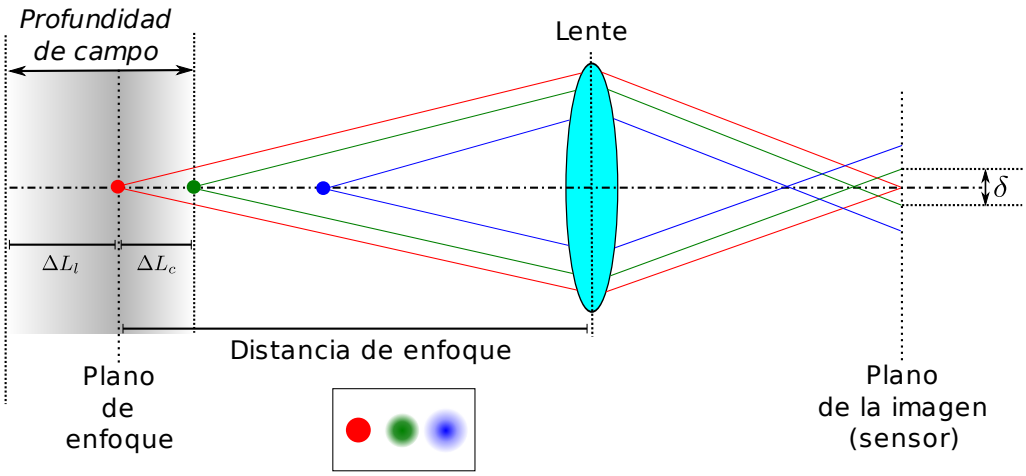


Figura 2.1: Parámetros involucrados en la profundidad de campo. Un objeto ubicado en el plano de enfoque aparecerá nítido en la imagen, mientras que un objeto situado al límite de la profundidad de campo aparecerá muy levemente desenfocado. Un objeto fuera de la profundidad de campo aparece totalmente desenfocado.

de campo deseada. Para esto será necesario tener en cuenta la iluminación, como se verá en la sección 2.3.

Los tres parámetros del sistema (la longitud focal f, la distancia de trabajo D_t y la profundidad de campo) generan un volumen en el espacio en el cual debe ubicarse la cabeza del usuario para poder realizar una captura con suficiente calidad para el reconocimiento. En la figura 2.2 se puede ver un esquema de este volumen, marcado en celeste.

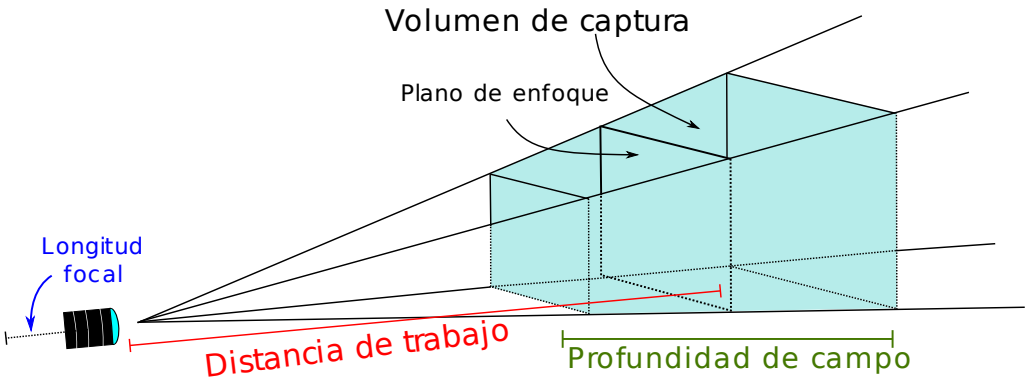


Figura 2.2: Volumen de captura generado por los parámetros del sistema

2.3. Iluminación

La iluminación juega un papel esencial en un sistema de reconocimiento de iris. La iluminación influye en el sistema de dos formas: en la intensidad de la luz y en el tipo de luz que se captura.

2.3.1. Intensidad de la luz

Como se dijo anteriormente, la profundidad de campo depende directamente de la apertura del diafragma. Es necesario también ajustar el diafragma de acuerdo a la cantidad de luz disponible. En el caso de que la iluminación sea débil, el diafragma deberá estar más abierto, con lo que se obtendrá una profundidad de campo pequeña. En forma inversa, cuando hay gran cantidad de luz disponible, se puede utilizar un diafragma más cerrado, obteniendo una profundidad de campo mayor. Es así como la intensidad de la luz disponible afecta en forma directa en el rango de trabajo del sistema.

2.3.2. Tipo de luz

Además de la intensidad de la luz, el tipo de luz que se utiliza es un factor determinante. Es necesario que la iluminación permita capturar la máxima información posible de la textura del iris.

El sistema de captura funciona básicamente capturando la luz reflejada por el iris en un sensor. Cada iris absorbe y refleja ciertas longitudes de onda dependiendo de la densidad celular y la pigmentación del estroma [BRM+06]. La luz reflejada es la que permite capturar la textura del tejido del iris, lo que posibilitará la identificación posterior.

Los ojos azules presentan una baja cantidad de melanina; en este caso la luz de alta longitud de onda (verde y rojo) es absorbida, mientras que la luz de baja longitud de onda (azul) es reflejada. En el caso de los ojos oscuros, la mayor parte de la luz visible es absorbida, lo que no permite distinguir con facilidad la textura del iris.

Luz infrarroja

Afortunadamente, si bien el iris puede absorber gran parte de la luz visible, en todos los casos refleja una cantidad considerable de luz infrarroja [Dau04]. Esto implica que se puede capturar la luz infrarroja reflejada por el iris para obtener imágenes con suficiente información.

En [HCTW06] se demuestra que el rango de longitud de onda con el cual se puede generar un buen contraste en las imágenes del iris ronda entre los 700 y 900 nanómetros (nm). Este rango de iluminación se corresponde con el infrarrojo cercano. En la figura 2.3 se puede ver una comparación de las distintas longitudes de onda que son reflejadas por los distintos colores de ojos. Se puede ver que la reflectancia del espectro visible (azul, rojo y verde) varía de acuerdo al color del ojo, pero el espectro infrarrojo tiene una alta reflectancia en todos los casos.

Para poder capturar imágenes en el infrarrojo cercano, es necesario contar con dos cosas:

- Una **cámara sensible al infrarrojo**. Los sensores CCD o CMOS de todas las cámaras son sensibles al infrarrojo cercano. Por lo general esto provoca problemas a la hora de capturar imágenes en el espectro visible, por lo que los fabricantes colocan un filtro enfrente del sensor que bloquea la luz infrarroja. Cualquier cámara puede convertirse en una cámara sensible al infrarrojo removiendo este filtro y reemplazándolo a su vez por un filtro que bloquee la luz visible (lo que ocasionaría problemas si se desea capturar únicamente imágenes en el infrarrojo). Otra posibilidad es conseguir directamente cámaras adaptadas para capturar imágenes en el infrarrojo.

- Una **fuente de iluminación infrarroja**. Esta fuente deberá iluminar al ojo del usuario, a fin de que la luz infrarroja reflejada por el iris sea capturada por la cámara. En el mercado es sencillo conseguir LEDs infrarrojos, por lo que se puede armar un dispositivo de iluminación a base de LEDs. Se puede regular la intensidad del iluminador infrarrojo variando la cantidad de LEDs que se utilicen.

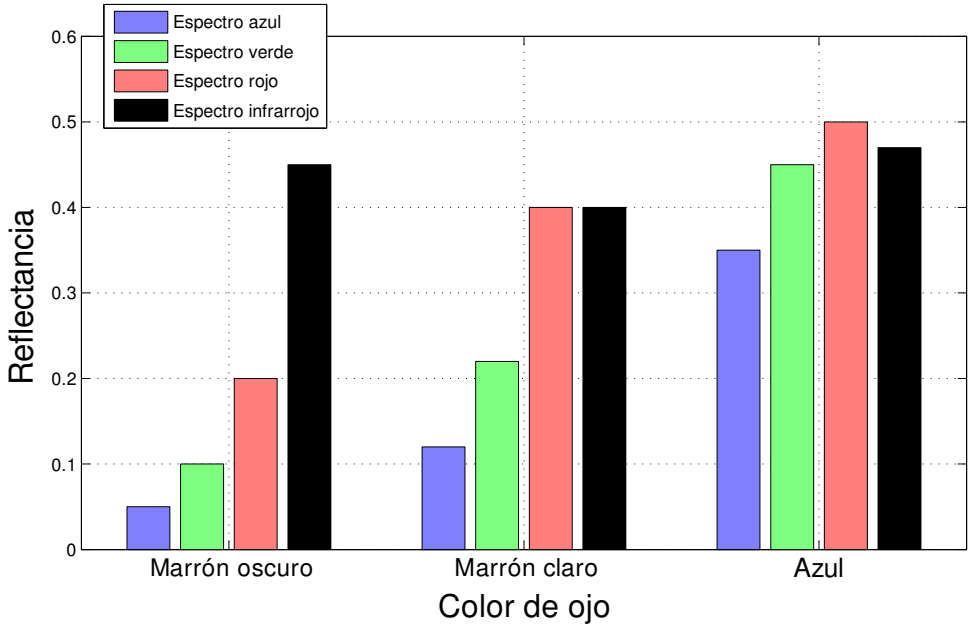
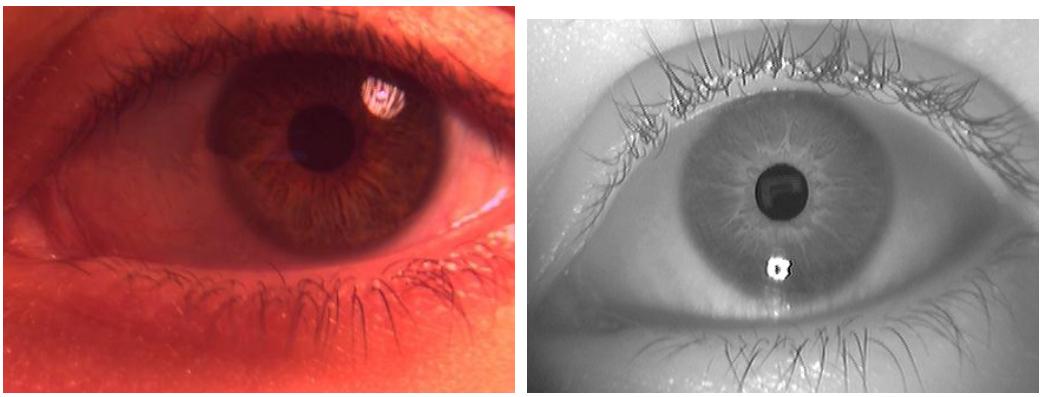


Figura 2.3: Valores de reflectancia cromática para distintos colores de iris. Adaptado de [BRM+06]. Se puede ver cómo la reflectancia del espectro infrarrojo es alta y constante para todos los colores de iris.

En la figura 2.4 se muestra el ejemplo de un ojo capturado con luz visible y el mismo ojo capturado con luz infrarroja. Se puede ver que en el caso de la luz visible hay bastante poco contraste en el área de la texutra del iris, mientras que con luz infrarroja el área del iris aparece mucho más clara y detallada (únicamente se presenta un reflejo especular producido por el iluminador infrarrojo).



(a) (b)

Figura 2.4: Ejemplo de un ojo capturado bajo dos condiciones de iluminación distintas. La figura 2.4a muestra una imagen capturada utilizando el espectro visible (iluminación mediante una lámpara incandescente). La figura 2.4b en cambio fue capturada en el espectro infrarrojo cercano, utilizando un iluminador de LEDs infrarrojos.

Una ventaja adicional de utilizar luz en el infrarrojo cercano es que la misma es invisible para el ojo humano, por lo que una fuente de iluminación en el infrarrojo no encandilará ni molestará al usuario, cosa que ocurriría si se usara una fuente de luz visible.

Sea cual fuere el tipo de iluminación que se elija, es necesario definir también la ubicación de

la fuente de luz. La córnea del ojo humano presenta reflejos especulares, y estos reflejos pueden invadir la región del iris en la imagen capturada.

Para solucionar esto aunque sea en forma parcial, es posible alinear la fuente de iluminación y el lente de la cámara en el mismo plano horizontal; esto causará que el reflejo de la luz se ubique justo en la pupila y no invada la parte del iris en la imagen. Para lograr esto, por lo general se utliza un anillo de LEDs ubicado alrededor del lente de la cámara. De no ser posible esto, la fuente de iluminación deberá ser colocada en alguna otra posición, siempre apuntando hacia la zona donde se ubicará el ojo del usuario. En este caso, el sistema deberá tener en cuenta los reflejos especulares causados por la iluminación en el momento de reconocer el iris del usuario

Luz multiespectral

En [BRM+06] se analiza la posibilidad de usar cuatro bandas de frecuencias (azul, verde, roja e infrarroja) al capturar imágenes del iris. Se demuestra que cada longitud de onda reflejada por el iris presenta diferentes componentes texturales, por lo que se deduce que si se utiliza la información presente en distintas longitudes de onda la identificación se hace más exacta que si se utiliza una única banda de frecuencias. Sin embargo, la captura de múltiples bandas aumenta la complejidad del sistema ya que se necesitan varios sensores, uno por cada banda.

2.4. Tipos de sistemas de captura

Hasta ahora se analizaron los distintos componentes físicos que componen un sistema de captura para reconocimiento de iris. Cada uno de estos componentes se puede adaptar para formar la configuración deseada (siempre y cuando se disponga de los componentes requeridos).

En cuanto a las configuraciones posibles, se pueden mencionar dos grandes categorías: los **sistemas de captura directa** y los **sistemas de captura no cooperativos**.

2.4.1. Sistemas de captura directa

Los sistemas de captura directa son los más comunes en el mercado. Su configuración es sencilla, consta de una única cámara dedicada a capturar imágenes del ojo del usuario.

Estos sistemas requieren una participación activa del usuario. El mismo debe posicionarse de acuerdo a algunos parámetros del sistema (como puede ser la distancia a la cámara, a una altura específica, etc.) y luego alinear su ojo con la cámara, a fin de que el plano del ojo quede paralelo al plano de la imagen. Luego, el usuario debe quedarse en una posición que permita capturar imágenes nítidas (sin movimiento) de su ojo.

Los sistemas de captura directa presentan dos grandes limitaciones:

- Poseen *parámetros fijos*: por lo general, los parámetros de estos sistemas (principalmente la distancia de trabajo) se encuentran prefijados. Esto hace que el sistema sea poco flexible frente a condiciones no previstas.
- Requieren una *participación activa del usuario*: como se dijo antes, es necesario que el usuario se posicione adecuadamente para que la cámara logre capturar una imagen de su ojo. En algunos casos, esto puede no ser deseable, como por ejemplo si el usuario no desea someterse al reconocimiento.

2.4.2. Sistemas de captura no cooperativos

Estos sistemas intentan no imponer condiciones a la captura de imágenes, o al menos reducirlas al mínimo. En [SCZY02] se mencionan los principales problemas a la hora de hacer más flexibles los sistemas de captura:

- *Iluminación*: En un entorno no controlado, el iris presentará reflejos de las luces ambiente que eliminarán los detalles en la textura del iris.
- *Ángulo de captura*: En los sistemas de captura, se requiere que el ojo esté alineado con la cámara a fin de que el iris sea un círculo (casi) perfecto. Si se elimina este requerimiento, el círculo del iris puede convertirse en una elipse, producto de la distorsión proyectiva del sistema de captura. Esta distorsión en la textura es difícil de controlar y puede impactar en el proceso de identificación.
- *Distancia de captura*: Si bien la distancia se puede agrandar utilizando lentes de mayor longitud focal, mientras mayor sea la misma más difícil será ubicar el ojo del usuario debido a la disminución del ángulo de captura. Además, en este caso, el sistema será mucho más susceptible a los movimientos de la cabeza del usuario, introduciendo *motion blur* y reduciendo la calidad de las imágenes.

Uno de los primeros sistemas de reconocimiento de iris que solucionó algunos de las limitaciones mencionadas fue el sistema "*Iris on the move*", introducido por Matey et al [MNH+06]. Una observación importante hecha en este trabajo es que las limitaciones de los sistemas existentes hasta el momento eran causadas únicamente por el sistema de captura, ya que los algoritmos utilizados funcionaban independientemente del sistema de captura utilizado.

En casos extremos, se puede mencionar el trabajo de Fancourt et al. [FBH+05], en el que mediante un complicado sistema logra capturar buenas imágenes a una distancia de 10 metros, aunque las condiciones de captura son bastante restringidas.

2.5. Sistemas de captura existentes

A continuación, se analizarán algunos sistemas de captura de iris implementados en distintos trabajos.

2.5.1. Daugman

El sistema propuesto por Daugman [Dau93] consta de una cámara de video que captura la luz reflejada por un *beam splitter* (espejo que refleja el 50 % de la luz, y el otro 50 % pasa a través de él). Posee una pantalla LCD que le permite al usuario ver en tiempo real lo que está capturando la cámara. Esto permite el correcto posicionamiento de la cabeza. La función del beam splitter es doble: permite que la cámara no se interponga entre el ojo del usuario y la pantalla LCD, y permite que el usuario puede ver lo proyectado en dicha pantalla.

Utiliza un potente lente de 330mm y la distancia de trabajo varía entre 15 y 46 centímetros [WAG+94]. En la figura 2.5 se puede ver un esquema del sistema propuesto.

2.5.2. Sarnoff

En [WAG+94] Wildes et al. describen un sistema completo de captura. En la figura 2.6 se puede ver un esquema del sistema propuesto. La particularidad de este sistema es que utiliza luz visible (no infrarroja), y utiliza dos marcos que permiten al usuario ubicarse en la posición correcta de captura. Utiliza un lente de 80mm, que permite una distancia de trabajo de 20cm. Para evitar los reflejos especulares de la luz en el ojo, utiliza primero un filtro difusor para que la luz sea distribuida de manera uniforme. En segundo lugar, utiliza un filtro para polarizar la luz en una dirección determinada, aprovechándose del hecho de que cuando la luz sea reflejada por el ojo el filtro polarizador bloqueará la mayor parte de la luz, dejando pasar sólo la luz difusa polarizada en la dirección definida por el filtro.

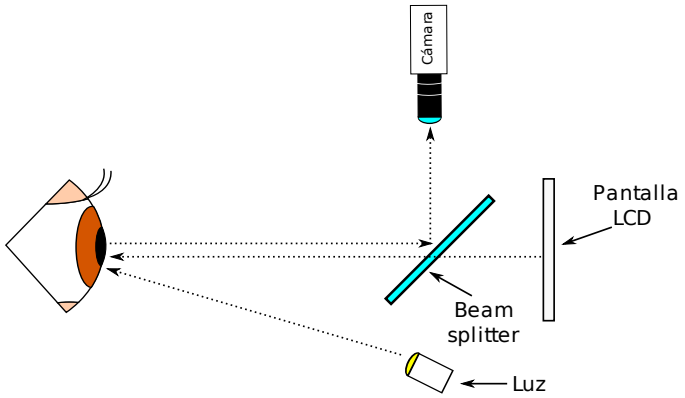


Figura 2.5: Sistema de captura propuesto por Daugman

Como el sistema utiliza luz visible, la intensidad de la luz no es muy alta o de lo contrario podría resultar molesto para el usuario. Por ello, debe usar una apertura de diafragma grande, lo que causa que la profunidad de campo del sistema sea pequeña, de aproximadamente 1 centímetro.

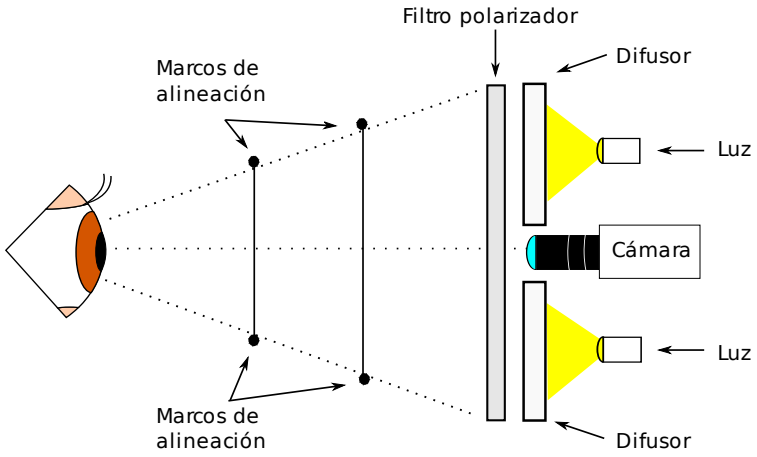


Figura 2.6: Sistema de captura propuesto por Wildes et al.

2.5.3. Iris on the move

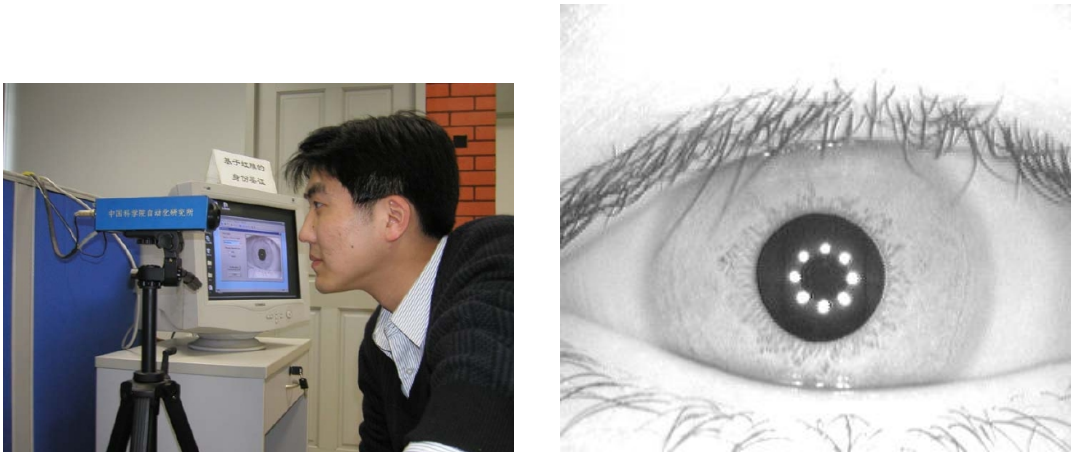
El sistema *Iris on the move*, también realizado en la corporación Sarnoff, consta de un conjunto de cámaras ubicadas en un arco por el cual el usuario debe pasar, a velocidad de paso normal, sin posicionarse de ninguna forma en particular. La única condición impuesta es que el usuario debe mirar a una de las cámaras. Esto no elimina el requerimiento de la participación activa del usuario, aunque la reduce al mínimo.

Este sistema hace uso de varias cámaras de muy alta resolución (2 megapixels) ubicadas a distintas alturas, lentes de alta longitud focal (210 mm) e iluminación estroboscópica (intermitente). Esto permite generar un cubo de captura en el espacio de 20 centímetros de largo, 37 centímetros de ancho y una profundidad de campo de entre 5 y 12 centímetros, a una distancia de 3 metros.

2.5.4. CASIA

El sistema CASIA utiliza un arreglo de LEDs infrarrojos ubicados alrededor de lente de la cámara, lo que ocasiona que los reflejos especulares se ubiquen justo en el área de la pupila [TZ]. En la figura 2.7 se puede ver una foto del sistema en uso, y una fotografía de un ojo capturada por el mismo sistema.

Las imágenes capturadas por este sistema tienen una resolución de 320x280 pixels, y el diámetro promedio del iris en dicha base es de aproximadamente 180 pixels.



(a) (b)

Figura 2.7: Sistema de captura propuesto por CASIA: 2.7a fotografía del sistema en uso, 2.7b fotografía del ojo del usuario capturada por el sistema. Se pueden ver claramente los reflejos especulares causados por el iluminador de LEDs.

2.6. Sistema implementado en este trabajo

El sistema implementado tiene los siguientes componentes:

- Una cámara del tipo CCTV PAL-N blanco y negro de alta sensibilidad, sensible al infrarrojo. Se eligió una cámara de video en vez de una cámara fotográfica ya que una cámara de video es la solución más práctica en un entorno en tiempo real.
- Un lente manual de tipo zoom con un rango focal de 6-60mm. [3]
- Un iluminador de LEDs infrarrojos, dispuestos en forma circular.

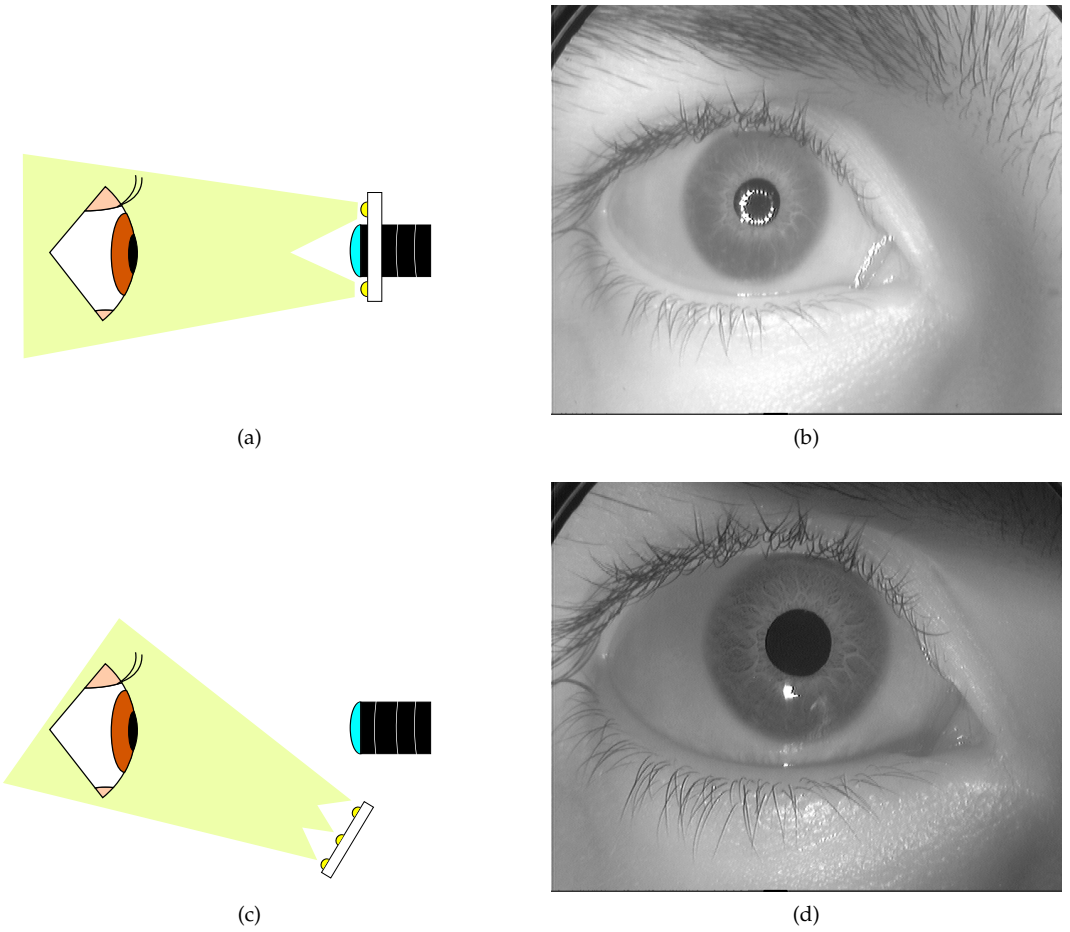
La cámara se monta sobre un trípode y, detrás de la cámara, se ubica un monitor de computadora que mostrará el video en tiempo real capturado por la cámara que le permitirá al usuario alinearse correctamente. La distancia de trabajo se ubica a unos 15 centímetros de la cámara. Esta distancia permitirá un radio del iris superior a los 200 pixels, que es justamente el diámetro mínimo establecido por el estándar ISO.

El arreglo de LEDs infrarrojos tiene una intensidad suficiente como para permitir una apertura de diafragma pequeña, lo que a su vez permite una profundidad de campo considerable, de aproximadamente 10 centímetros. Se hicieron varias pruebas con el fin de determinar la mejor ubicación del iluminador infrarrojo; estas pruebas serán descriptas a continuación.

[3] Un lente de 60mm no es óptimo para capturar imágenes del ojo ya que requiere que el usuario esté a una distancia muy corta, pero era el único lente disponible en el mercado en el momento de implementar el sistema.

2.6.1. Pruebas realizadas

A fin de ver cual es la mejor ubicación para iluminador infrarrojo, se hicieron varias pruebas. En la figura 2.8 se muestran dos pruebas realizadas sobre el mismo ojo: en una, se ubica el iluminador infrarrojo alrededor del lente de la cámara, y, en la otra, debajo de la cámara. Se puede ver claramente cómo en el primer caso los reflejos especulares están ubicados dentro del área de la pupila, mientras que en el segundo caso el reflejo invade parte del área del iris, aunque el tamaño de dicho reflejo es muy pequeño en relación con el área total.



(a) (b)

(c) (d)

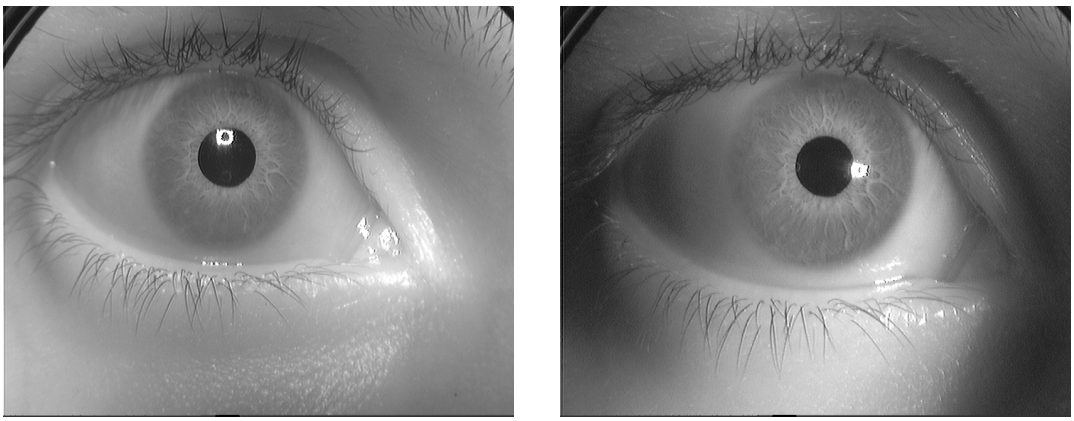
Figura 2.8: Dos esquemas de iluminación: 2.8a iluminación desde la cámara, 2.8c iluminación inferior, 2.8b, 2.8d sus respectivos resultados

Otras pruebas realizadas fueron iluminar el ojo desde la posición superior y desde el costado (figura 2.9). En todos los casos se obtuvieron resultados muy similares en cuanto a contraste y nivel de detalle (aunque en el caso de la iluminación lateral se produjeron fuertes sombras). Esto lleva a razonar que la ubicación de la fuente de iluminación es un parámetro muy flexible del sistema, y que no tiene mayores impactos en el funcionamiento del mismo.

Finalmente, se decidió ubicar el iluminador infrarrojo en la parte inferior de la cámara por tres motivos:

1. Permite que la pupila sea completamente negra. Esto servirá a la hora de desarrollar un algoritmo rápido de segmentación, como se verá en el capítulo 3.

2. Como la distancia focal del lente no es muy grande, si el usuario se acerca demasiado al lente



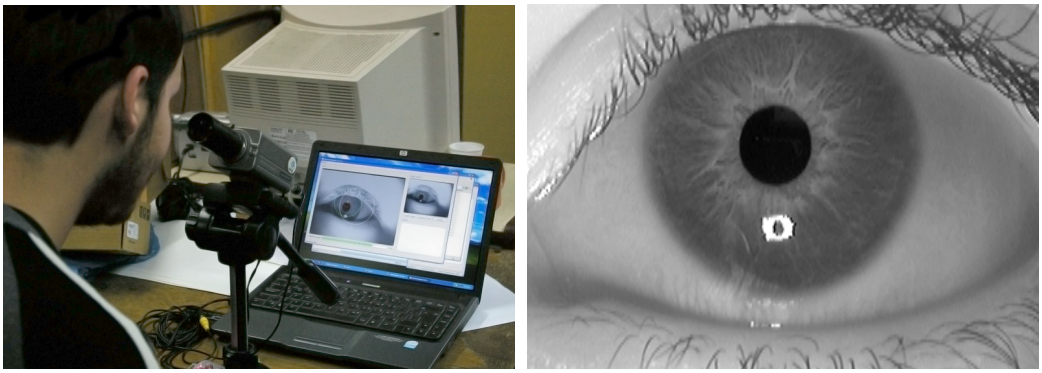
(a) (b)

Figura 2.9: 2.9a Iluminación superior, 2.9b desde la derecha

y el iluminador está alrededor del mismo, se proyectan sombras que justamente afectan el área del iris.

3. La textura del iris parece tener un mayor contraste al ser iluminada desde abajo que desde otras direcciones.

En la figura 2.10 se puede ver el sistema implementado junto con una imagen capturada.



(a) (b)

Figura 2.10: Sistema de captura implementado. La figura 2.10b muestra una fotografía del sistema. La cámara está montada sobre un trípode, y tiene una inclinación que permite una posición más cómoda al usuario. El iluminador infrarrojo se encuentra en la parte inferior de la imagen. La cámara está alineada con el monitor de la computadora, que permite al usuario ver en tiempo real las imágenes capturadas por el sistema. La figura 2.10b muestra una imagen típica obtenida por el sistema de captura, que posee un diámetro de 215 pixels y un nivel de detalle considerable. Se puede ver el reflejo del iluminador infrarrojo en la parte inferior.

2.7. Resumen

En este capítulo se detallaron los distintos factores que influyen a la hora de implementar un sistema de captura de imágenes de iris. Se analizaron los dos principales problemas, que son

capturar imágenes en foco y al mismo tiempo ser flexible y poco intrusivo para el usuario. Se mostró también cómo pueden interactuar los distintos componentes del sistema de captura en sistemas utilizados en la práctica. Finalmente, se describió el sistema de captura implementado.

Capítulo 3

Segmentación

La *segmentación* de la imagen del iris consiste en encontrar los parámetros que determinan la ubicación, forma y tamaño del iris dentro de la imagen capturada. Con el fin de analizar la textura del iris humano automáticamente, es necesario identificar en la imagen la región que contiene el iris y separarla de otros elementos de la imagen que no contienen información relevante, como la pupila, párpados y pestañas.

En este capítulo se analizarán algunos métodos de segmentación existentes y se propondrá un nuevo algoritmo de segmentación optimizado para funcionar en tiempo real.

3.1. Segmentación del iris y la pupila

Existe mucha bibliografía respecto a la segmentación de imágenes para sistemas de reconocimiento de iris. Un algoritmo de segmentación puede estar diseñado para ser utilizado bajo muchas condiciones de captura o puede estar optimizado para un conjunto controlado de condiciones. En este último caso es necesario hacer algunas suposiciones que deberán mantenerse constantes a lo largo de todo el proceso de captura, o de lo contrario los resultados no serán buenos. Por ejemplo, un algoritmo que asuma que se segmentarán imágenes capturadas en el infrarrojo, donde existe un buen contraste entre la pupila y el resto de la imagen no funcionará necesariamente con imágenes capturadas con luz natural, donde el contraste no es tan pronunciado.

3.1.1. Modelando el iris y la pupila

Para localizar el iris y la pupila, es necesario modelar ambos elementos de alguna forma. Lo más común es representar tanto el iris como la pupila como dos círculos casi concéntricos, con lo cual el resultado de la segmentación serán los parámetros de ambos círculos. Un círculo cualquiera puede ser expresado con tres parámetros: las coordenadas (x_0, y_0) de su centro y su radio r, así que la segmentación deberá encontrar un total de seis parámetros: tres para el círculo de la pupila y tres para el círculo del iris.

También se puede representar al iris y la pupila como dos elipses [MIA05], con lo cual la cantidad de parámetros a encontrar asciende a 10 ya que cada elipse se define mediante 5 parámetros (dos longitudes para sus ejes, el centro de la elipse y el ángulo de rotación). Esto provee una representación más exacta para el iris y la pupila, pero al ser necesario encontrar más parámetros, los tiempos de ejecución de los algoritmos aumentan notablemente. Además, la ventaja de utilizar elipses en vez de círculos para representación no resulta ser muy significativa, ya que tanto el iris como la pupila son casi perfectamente circulares.

Interesa entonces el problema de detectar círculos en la imagen. Para esto, existe principalmente dos algoritmos utilizados por los sistemas de reconocimento de iris: el operador integro-diferencial y la detección por transformada de Hough.

Operador integro-diferencial

El operador integro-diferencial fue propuesto por Daugman [Dau93] en su sistema original de reconocimiento de iris. La base del operador consiste en detectar cambios bruscos en el tono de gris a lo largo de circunferencias en la imagen, con la idea de que un cambio brusco se traduce en un máximo en la derivada de los tonos de gris de las circunferencias centradas en el punto (x_0, y_0).

La expresión del operador para encontar el círculo de centro (x_0, y_0) y radio r en la imagen I es la siguiente:

$$\arg\max_{x_0,y_0,r} \left| G_\sigma(r) * \frac{\partial}{\partial r} \oint_{x_o,y_o,r} \frac{I(x,y)}{2\pi r} \mathrm{d}s \right| \tag{3.1}$$

donde

- s representa el contorno sobre el cual se hace la integración, en este caso, un contorno circular
- $G_\sigma(r)$ es una función Gaussiana que define el nivel de detalle en el que se trabaja: mientras más precisión se desea, menor será el parámetro de suavización σ.

Este operador funciona iterativamente, buscando primero una aproximación y luego obteniendo resultados más precisos. La cantidad de niveles de detalle puede ser ajustada para que el operador trabaje en forma más veloz.

Una propiedad importante de este operador es que se puede generalizar para detectar otros tipos de curvas parametrizables. Cambiando el contorno s de una circunferencia a una parábola, es posible construir de manera similar un detector de parábolas, que luego puede ser utilizado para detectar los parámetros de los párpados.

Transformada circular de Hough

La mayoría de los trabajos utilizan la transformada circular de Hough para detectar los parámetros de la pupila y del iris en forma simultánea ([WAG+94, MTWZ03, Mas03]).

Este método es común en el ámbito de procesamiento de imágenes principalmente para encontrar curvas parametrizables en imágenes. Se comienza creando una imagen binarizada, B,, aplicando un detector de bordes a la imagen original. Luego, cada punto perteneciente a un borde suma "votos" en cada curva que pase por dicho punto.

Para segmentar el iris o la pupila, se suele utilizar el detector de bordes de Canny, y a veces el mismo es ajustado para detectar los bordes verticales. Esto tiene una doble ventaja: por un lado, los bordes horizontales del iris suelen estar ocultos por los párpados, y por otro lado, los bordes de los párpados no aparecen en la imagen resultante ya que es de esperar que los mismos aparezcan en forma horizontal en la imagen. Se define entonces la función C como:

$$C(x_c, y_c, x, y, r) = \begin{cases} 1 & \text{Si la circunferencia de radio } r \text{ y centro} \\ & (x_c, y_c) \text{ pasa por el pixel } (x, y) \\ 0 & \text{Si no} \end{cases} \tag{3.2}$$

Una vez calculados los bordes, se crea un espacio discretizado tridimensional H que va a acumular los votos para cada uno de los tres parámetros (x_c, y_c, r) del círculo. Así, se define H de la siguiente manera:

$$H(x, y, r) = \sum_{(x_c,y_c)\in B} C(x_c, y_c, x, y, r) \tag{3.3}$$

Finalmente, los parámetros del círculo están dados por:

$$(x_c, y_c, r) = \arg\max_{(x_c, y_c, r)} H(x_c, y_c, r) \tag{3.4}$$

En forma similar, la transformada de Hough se puede modificar fácilmente para detectar parábolas en vez de círculos, con el fin de segmentar los párpados.

Sin embargo, este método presenta algunos problemas:

1. El resultado de la transformada de Hough está condicionado por los parámetros del algoritmo de detección de bordes. En el caso de las imágenes del iris, es necesario ajustar los parámetros de este algoritmo de forma tal que los círculos de la pupila y el iris se encuentren bien marcados en la imagen resultante.

2. Dichos parámetros son poco flexibles: si existe una modificación en las condiciones de captura, el detector de bordes puede dar muchos falsos positivos. Con parámetros incorrectos es normal que la imagen resultante o bien no contenga ningún borde útil, o bien, más comunmente, que se encuentren demasiados bordes en la imagen. Esto último es producto de la naturaleza misma del iris, que tiene patrones complicados que pueden aparecer en el algoritmo de detección de bordes.

3. El tiempo de procesamiento del algoritmo es directamente proporcional a la cantidad de puntos de bordes detectado en la imagen. Si la imagen tiene muchos bordes, como suelen aparecer en las pestañas y los párpados cuando la imagen está bien enfocada, el algoritmo resulta muy lento.

4. Una imagen con demasiados bordes ocasiona tanto una degradación en el tiempo de ejecución de la transformada de Hough como en el resultado de la misma (es posible que se encuentren "círculos" que en realidad no existen, por lo que es necesario recurrir a heurísticas que permitan comprobar si un círculo detectado realmente se corresponde con la pupila o el iris).

3.1.2. Segmentación de pestañas

Existen también trabajos que tratan el problema de detectar las pestañas. Kong y Zhang [KZ01] proponen segmentar cada pestaña. Para esto, dividen las pestañas en dos clases: las *separables* y las *múltiples*. Las pestañas separables se pueden identificar individualmente, mientras que las múltiples son aquellas que se superponen en un área pequeña y no se pueden distinguir.

Para segmentar las pestañas separables se calcula la convolución de la imagen con un filtro de Gabor diseñado especialmente[1], mientras que las pestañas múltiples se segmentan simplemente buscando porciones de la imagen con varianza pequeña en los tonos de gris de dicha porción. Además, se agrega una condición de *conectividad*: las pestañas deben ser componentes conexas, por lo que si un pixel aislado fue clasificado como pestaña, es eliminado.

3.2. Solución propuesta

La solución implementada en este sistema está diseñada con el fin de obtener una segmentación que permita el procesamiento de las imágenes en tiempo real. Esto implica que el tiempo de segmentación debe ser el menor posible, de aproximadamente 100 milésimas de segundo por imagen, lo que permite procesar el video a una tasa de aproximadamente 10 o más frames por segundo.

[1] En el capítulo 5 se hablará de los filtros de Gabor en profundidad

La idea no fue implementar un algoritmo que funcione en todos los casos sino que pueda funcionar *bien* bajo las condiciones del sistema de captura, y que permita cierta flexibilidad en los parámetros y un grado aceptable de robustez.

Un algoritmo robusto de segmentación debe tener en cuenta que las características de la pupila y del iris son distintas:

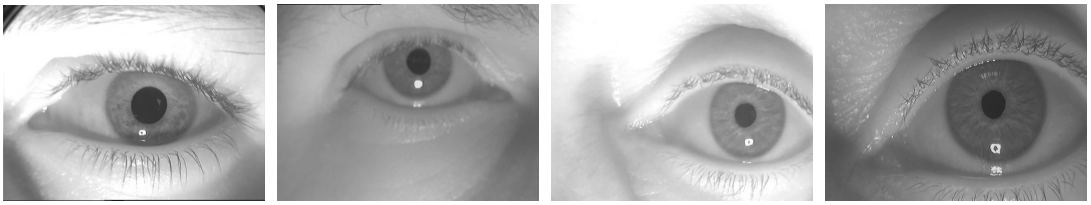
- La pupila suele tener un nivel de iluminación muy bajo, casi negro
- El iris es más claro que la pupila, pero menos claro que la esclera, y por lo general se encuentra parcialmente oculto por los párpados y las pestañas.

Como la pupila suele ser un círculo bien distinguible bajo casi todas las condiciones, una buena heurística es localizar primero la pupila y a partir de ahí localizar el iris, ya que el mismo es un círculo casi concéntrico a la pupila.

Por lo tanto, la segmentación del iris se realiza en dos pasos: en el primero, se detecta la ubicación de la pupila en la imagen, y en la segunda se parte de la posición de la pupila para encontrar los parámetros del iris.

3.2.1. Segmentación de la pupila

La pupila se distingue fácilmente en la imagen como un círculo negro bajo las condiciones de iluminación predefinidas, descriptas en el capítulo 2. En la figura 3.1 se pueden ver algunas imágenes obtenidas por el sistema de captura bajo distintas condiciones de iluminación y distancia de trabajo.



(a) (b) (c) (d)

Figura 3.1: Capturas típicas realizadas por el sistema de captura. Las imágenes 3.1c y 3.1d corresponden al mismo ojo. Se puede ver cómo varía el tamaño del ojo en función de la distancia a la cámara, y el diámetro de la pupila por motivos naturales.

Preprocesamiento

Si bien lo ideal sería mantener todos los parámetros de captura constantes, es más realista suponer que pueden existir variaciones en las situaciones de captura, y se debe permitir al algoritmo de segmentación cierta flexibilidad a la hora de procesar las imágenes.

Observando las imágenes, se pueden ver algunas cosas:

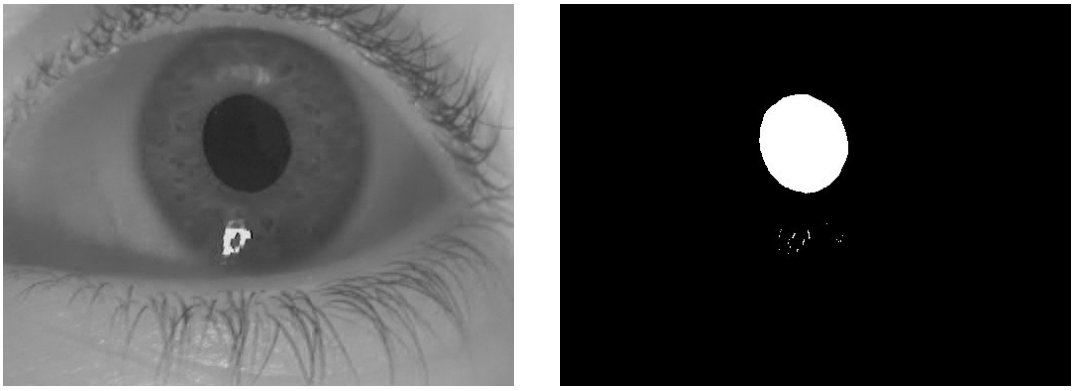
- El reflejo especular del iluminador infrarrojo aparece en la parte inferior del ojo. En el capítulo 2 se concluyó que esto permite un mayor contraste en la textura del iris.
- El nivel de iluminación no es constante, y depende de la intensidad del iluminador infrarrojo y de la apertura del diafragma del lente, como se vio en el capítulo 2
- Lo mismo ocurre con el tamaño del ojo en la imagen. Varía en forma directamente proporcional a la distancia al lente (es decir, mientras más cerca está el ojo del lente, más grande aparece). Aunque la distancia de trabajo es constante, es de esperar que existan variaciones en la distancia y que el tamaño del ojo no sea siempre constante.

- Si bien la intensidad de la iluminación es variable, la pupila siempre aparece como la característica más oscura de toda la imagen. Como el reflejo del iluminador se encuentra en la parte inferior, dicho reflejo no aparece *dentro* de la pupila. Esto depende del sistema de captura y no se cumple en todos los casos; por ejemplo en la base de datos CASIA3 el iluminador infrarrojo se encuentra alrededor del lente por lo que los reflejos se encuentran siempre dentro de la pupila.

La pupila es la característica más fácilmente distinguible de la imagen debido a su gran contraste respecto al resto de la imagen. Además, dadas las características del ojo, el iris siempre se encuentra alrededor de la pupila. Por estos dos motivos, es razonable iniciar el algoritmo de segmentación del iris ubicando la pupila en la imagen y luego detectar la posición del iris, partiendo de la ubicación de la pupila.

Para detectar rápidamente la pupila se puede usar alguno de los métodos descriptos anteriormente. El operador integro-diferencial es particularmente bueno ya que el círculo de la pupila está bien marcado, y dicho operador puede ser fácilmente optimizado, como se verá a continuación.

Este operador puede presentar ciertos problemas, como por ejemplo detectar erróneamente el contorno entre el iris y la esclera como si fuera el contorno entre la pupila y el iris. Para solucionar esto, es posible filtrar la imagen de forma tal que sólo se consideren los puntos de la pupila. Por ejemplo, aplicando un valor de threshold adecuado, se puede generar una imagen binaria en la cual sólo el círculo de la pupila se encuentra presente, además de un poco de ruido producido por las pestañas, cejas y otros elementos (figura 3.2).



(a) (b)

Figura 3.2: Imagen binarizada luego de aplicar un threshold manual. La pupila se extrae en forma casi perfecta.

Sin embargo, las variaciones en la iluminación no permiten utilizar un valor único de threshold para todas las imágenes.

Para compensar la variación de iluminación, se realiza una ecualización de histograma a la imagen. Esto va a permitir que la distribución de grises sea uniforme a lo largo de la imagen, y, como la pupila siempre es la característica más oscura, los niveles de grises correspondientes a la pupila luego de la ecualización serán muy próximos a 0.

Aún de esa forma, no existe un único valor de threshold adecuado para todas las imágenes. Además, el tono de gris de la pupila no es necesariamente uniforme ya que pueden existir reflejos. Por esto, no es conveniente tratar de binarizar la imagen. Un mejor enfoque consiste en elegir un rango de valores entre los cuales es probable encontrar los tonos de gris de la pupila.

De esta forma, se transforma el valor de gris de cada pixel de la imagen de acuerdo a la sigu-

iente fórmula:

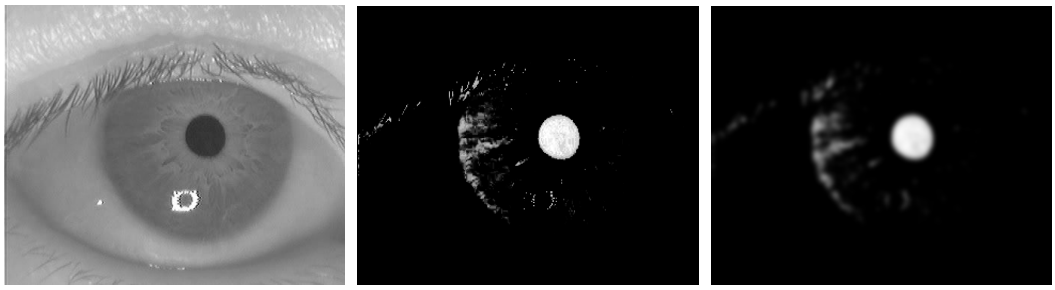
$$I'(x,y) = \exp\left(-\frac{(I(x,y)-\mu_p)^2}{2\sigma_p^2}\right) \tag{3.5}$$

donde

- I' es la imagen resultante de aplicar la transformación
- μ_p es el valor promedio de gris que se espera encontrar para la pupila
- σ_p es el desvío esperado para el promedio de gris de la pupila

Esta transformación da como resultado otra imagen donde la intensidad de cada pixel es proporcional a la probabilidad de que el nivel de gris en la imagen original se corresponda con el nivel de gris de la pupila.

En la figura 3.3 se puede ver el resultado de aplicar este filtro a la imagen. Experimentalmente se encontró que para el sistema de captura utilizado los mejores valores de los parámetros son $\mu_p = 0$ y $\sigma_p = 5$, lo que implica como se dijo antes que la pupila es la característica más oscura de la imagen. Además de la pupila, luego de aplicar el filtro puede aparecer en el resultado ruido como los párpados, sombras o inclusive otras partes más oscuras del iris. Esto no presenta un gran problema, ya que el objetivo es aislar el círculo de la pupila para optimizar el resultado del operador integro-diferencial, y se puede ver en dicha figura que la pupila es la característica más importante del resultado. Aún así, para que la operación sea más resistente al ruido, se le aplica posteriormente un desenfoque gaussiano (figura 3.3c) que elimina gran parte del ruido.



(a) (b) (c)

Figura 3.3: Filtrado de la imagen. Al aplicar la transformación descripta, se descarta la mayor parte de la imagen que no se corresponde con la pupila. En 3.3c se aplicó un blur Gaussiano para eliminar los pixeles aislados y el ruido en general.

Operador integro-diferencial optimizado

Como se dijo anteriormente, el operador integro-diferencial sirve para encontrar bordes de formas parametrizables, localizando la posición de máxima variación de la intensidad dentro de la imagen. En el caso de la imagen filtrada de la forma que se explicó, el círculo de la pupila será la característica más notable, por lo que el operador tendrá un buen resultado.

La base del operador consiste en calcular el promedio de la imagen a lo largo de una circunferencia de centro (x,y) y radio r:

$$P(x,y,r) = \frac{1}{n}\sum_{k=0}^{n-1} I(x_k, y_k) \tag{3.6}$$

donde

$$x_k = x + r \cdot \cos\left(\frac{2\pi k}{n}\right) \tag{3.7}$$

$$y_k = y + r \cdot \sin\left(\frac{2\pi k}{n}\right) \tag{3.8}$$

Para un centro (x_c, y_c) dado, se busca el radio donde el crecimiento de P es máximo buscando el máximo en su derivada:

$$M(x_c, y_c) = \arg\max_r \left| \frac{\partial P}{\partial r}(x_c, y_c, r) \right|$$

donde M es una matriz que almacena el radio donde se encuentra la máxima variación en el tono de gris para las circunferencias centradas en el punto (x_c, y_c).

Entonces, buscando el punto (x_c, y_c) donde M se maximiza, se podrán encontrar los tres parámetros necesarios para identificar la circunferencia del círculo *mejor definido* de la imagen, que en este caso se corresponde con el círculo de la pupila (figura 3.4).

El problema de este operador es que, tal como está definido, requiere un gran número de operaciones (es decir, recorrer todos los pixels de la imagen computando los promedios de todos los círculos alrededor de dichos pixels), y esto es impráctico en un entorno en tiempo real.

Para acelerar el operador, se realizan dos modificaciones: búsqueda en cascada y cálculo del promedio del círculo mediante el algoritmo de Bresenham.

Búsqueda en cascada

La búsqueda en cascada consiste en aplicar el operador en varios niveles de detalle, donde el resultado de cada nivel es una estimación de la posición real del círculo. Cada nivel de detalle está definido por los parámetros $(\Delta x, \Delta y)$ y Δr, donde $(\Delta x, \Delta y)$ define la separación con la que se tomarán los centros y Δr es la variación del radio.

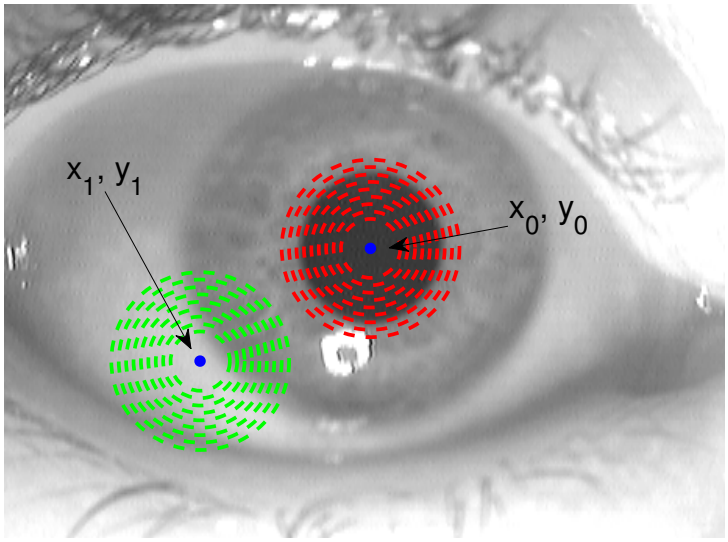
Esto funciona por la siguiente propiedad: si el círculo verdadero de la pupila está en la posición (x_c, y_c) y tiene radio r_c, es de esperar que para un valor Δr, se verifique que el valor de $|P(x_c, y_c, r_c - \Delta r) - P(x_c, y_c, r_c + \Delta r)|$ sea grande. Además, para un punto (x, y) cercano a (x_c, y_c), también se produce un salto en los tonos de grises de las circunferencias de radio cercanos a r_c.

Una vez que se hace la estimación inicial de la posición de la circunferencia, se repite el procedimiento reduciendo los valores de Δx, Δy y Δr (es decir, en una escala más fina) pero buscando únicamente en un entorno centrado en la posición estimada. La escala se va reduciendo cada vez más hasta llegar a $\Delta x = \Delta y = \Delta r = 1$, donde se encuentran los parámetros finales de la circunferencia con precisión de un pixel.

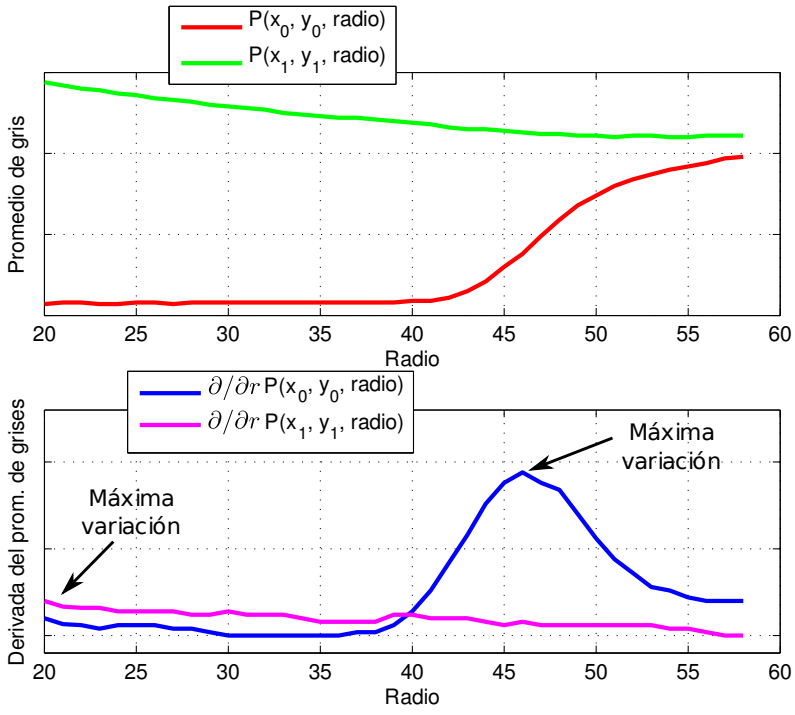
Algoritmo de Bresenham

El algoritmo de Bresenham [Fol90] es utilizado en el área de gráficos por computadora para dibujar rápidamente rectas, círculos y otras figuras geométricas simples. Permite generar las coordenadas de los pixels que componen la figura de a uno y sin repetir posiciones. Además, está diseñado para utilizar únicamente aritmética entera, por lo que es sumamente eficiente.

En el planteo original del operador integro-diferencial, el promedio de las circunferencias se calcula variando el ángulo θ respecto al centro de la circunferencia. Esto puede producir que en círculos pequeños se compute el mismo tono de gris dos veces y que en círculos grandes se salteen algunos valores. En ambos casos, esto es poco deseable. El algoritmo de Bresenham se puede modificar fácilmente para calcular dicho promedio.



(a)



(b)

Figura 3.4: En 3.4a se puede ver el funcionamiento del operador integro-diferencial para dos puntos de la imagen: el punto central de la pupila y un punto cualquiera (en rojo y verde, respectivamente). En 3.4b se ve los resultados para los promedios de las circunferencias centradas en dichos puntos variando el radio (gráfico superior) y la derivada de los promedios (gráfico inferior). Se puede ver que la derivada tiene una muy alta respuesta cuando la circunferencia se corresponde con el contorno de la pupila, representado por el valor máximo de la curva azul.

Implementación

La implementación de la segmentación de la pupila en este trabajo consiste en la búsqueda de los parámetros por medio del operador integro-diferencial de varios niveles calculado utilizando

el algoritmo de Bresenham. El pseudocódigo de estos métodos se puede ver en los algoritmos 3.1 y 3.2.

En el algoritmo 3.1, para la primera iteración, los valores de $r_{\text{mín}}$ y $r_{\text{máx}}$ definen el tamaño mínimo y máximo a buscar de la circunferencia de la pupila y depende exclusivamente de la resolución del sistema de captura y de la distancia de trabajo, como se explicó en el capítulo 2.

Los valores de Δx_n, Δy_n y Δr_n para $n = 1 \dots k$ son los valores que definen la resolución de cada nivel de detalle, y se debe cumplir que $\Delta x_k = \Delta y_k = \Delta r_k = 1$ al término del algoritmo, a fin de encontrar la ubicación de la circunferencia con resolución de un pixel.

3.2.2. Ajuste del contorno de la pupila

El algoritmo descripto anteriormente da como resultado una circunferencia que *aproxima* el contorno del iris. Sin embargo, el mismo no es necesariamente una circunferencia perfecta y suele tener desviaciones. Si no se tienen en cuenta estas desviaciones, se estará considerando parte de la pupila como el iris o se excluirá parte del iris. En la figura 3.5 se puede ver una ilustración de esto.

Es necesario entonces hacer una segmentación más fina del contorno de la pupila. Si bien en la mayoría de los trabajos se asume un contorno circular para la pupila, sólo trabajos recientes van más allá y utilizan técnicas basadas en contornos activos [AT06, HTS06] y snakes [Dau07] para modelar la pupila.

Utilizando un contorno, la pupila se puede representar como una función $\mathcal{C} : [0, 2\pi) \rightarrow \mathbb{R}^2$ donde, para un ángulo dado, se obtiene un punto de la imagen que representa el punto del contorno para ese ángulo. Luego, el objetivo de este paso es encontrar un conjunto de puntos que permita representar esta función.

Habiendo obtenido la circunferencia aproximada de la pupila, se puede expresar esta circunferencia como un contorno de acuerdo a la expresión anterior como:

$$\mathcal{C}_{\text{circ}}(\theta) = (x_{\text{pupila}} + r_{\text{pupila}} \cos\theta, y_{\text{pupila}} + r_{\text{pupila}} \sin\theta) \tag{3.9}$$

Esta es una muy buena aproximación inicial para el contorno. Por lo tanto, es el punto de partida para un algoritmo de segmentación mejorado.

El algoritmo implementado en este trabajo está basado en el método presentado recientemente por Daugman en [Dau07], que consiste en analizar una "corona" de la imagen centrada en el centro de la pupila. Para simplificar los cálculos, la corona se puede representar en coordenadas polares de la siguiente forma: se crea una imagen rectangular donde cada fila representa una circunferencia de la imagen centrada en el centro de la pupila, y cada columna un ángulo de cada circunferencia (figura 3.6):

$$I_p(\theta, \rho) = I\left(x_{\text{pupila}} + \rho \cdot \cos\theta, y_{\text{pupila}} + \rho \cdot \sin\theta\right) \tag{3.10}$$

con $\theta \in [0 \dots 2\pi)$ y $\rho \in [r_{\text{mín}} \dots r_{\text{máx}}]$, donde $r_{\text{mín}}$ y $r_{\text{máx}}$ son los radios mínimos y máximos del aro a extraer (se debe cumplir que $r_{\text{mín}} < r_{\text{pupila}} < r_{\text{max}}$, de modo que la circunferencia encontrada sea representada por una recta en I_p). En la figura 3.6b se puede ver el resultado de hacer esta transformación.

Al generar la imagen de esa manera, el problema de encontrar una serie de puntos en dos dimensiones que represente el contorno de la pupila se transforma a encontrar una curva unidimensional en la imagen en coordenadas polares. Se puede ver que mediante esta transformación el círculo de la pupila se convierte en una recta en el nuevo sistema de coordenadas (marcada con una línea roja en la figura 3.6b). La frontera entre la pupila y el iris es visible claramente en esta imagen como una curva próxima a la recta que representa el círculo de la pupila. Si la pupila fuera perfectamente circular, este contorno sería igual a dicha recta; sin embargo, se pueden apreciar zonas donde la pupila está por debajo de la recta y otras donde está encima.

Algoritmo 3.1: Operador integro-diferencial optimizado

Input: I: imagen
$x_{\text{mín}} = y_{\text{mín}} = 0$
$x_{\text{máx}} = \text{width}$
$y_{\text{máx}} = \text{height}$

Para $n = 1, \ldots,$ cant. de niveles de detalle
  Para $x = x_{\text{mín}}$ y mientras $x < x_{\text{máx}}$
    Para $y = y_{\text{mín}}$ y mientras $y < y_{\text{máx}}$
      Para $r = r_{\text{mín}}$ y mientras $r < r_{\text{máx}}$
        $P(x, y, r) = \text{PromedioCirculo}(I, x, y, r)$
        $r = r + \Delta r_n$
      Fin
      $y = y + \Delta y_n$
    Fin
    $x = x + \Delta x_n$
  Fin
  $(x_n, y_n, r_n) = \arg\max_{x,y,r} \left| \frac{\partial P}{\partial r}(x, y, r) \right|$
  $x_{\text{mín}} = x_n - \Delta x_n, x_{\text{máx}} = x_n + \Delta x_n$
  $y_{\text{mín}} = y_n - \Delta y_n, y_{\text{máx}} = y_n + \Delta y_n$
  $r_{\text{mín}} = r_n - \Delta r_n, r_{\text{máx}} = r_n + \Delta r_n$
Fin

Algoritmo 3.2: PromedioCirculo(I, x_c, y_c, r_c)

Input: I: imagen, (x_c, y_c): centro del circulo, r_c: radio del circulo
$x = 0, y = y_c, d = 3 - 2 \cdot r_c$
$n = S = 0$
$i = 0$

Mientras $x < y$
  $S = S + I(xc + x, yc + y) + I(xc - x, yc + y) + I(xc + x, yc - y) +$
    $+ I(xc - x, yc - y) + I(xc + y, yc + x) + I(xc - y, yc + x) +$
    $+ I(xc + y, yc - x) + I(xc - y, yc - x)$

  Si $d < 0$
    $d = d + 4x + 6$
  Si $d \geq 0$
    $d = d + 4(x - y) + 10$
    $y = y - 1$
  Fin

  $n = n + 8$
  $x = x + 1$
Fin

Retornar S/n

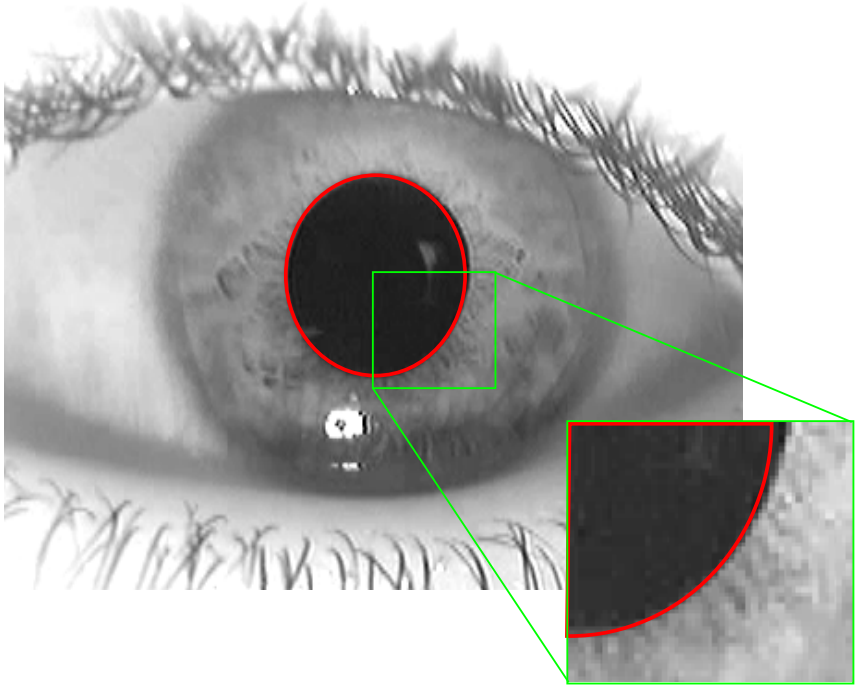
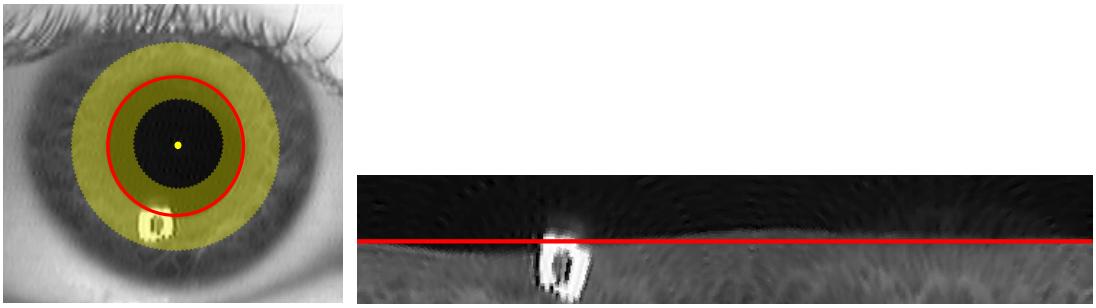


Figura 3.5: Efecto de considerar a la pupila como un círculo: en algunos casos, la pupila no es perfectamente circular, por lo que al utilizar un modelo circular de segmentación algunas partes de la pupila son consideradas parte del iris.



(a) (b)

Figura 3.6: Extracción de un aro de la imagen y conversión a coordenadas polares: En 3.6a se puede ver en amarillo el aro de la imagen a extraer, rodeando el círculo de la pupila (en rojo). La figura 3.6b muestra el aro convertido a coordenadas polares. El círculo de la pupila encontrado en el paso anterior aparece representado en esta figura como una recta.

Ya que el contorno entre la pupila y el iris está bien contrastado, es de esperar que la derivada en la dirección radial de la imagen (o, equivalentemente, la derivada en la dirección y en la imagen transformada) tenga altos valores absolutos en dicho contorno. En la figura 3.7 se ve la derivada de la imagen y el contorno bien definido.

Se puede definir entonces el nuevo contorno aproximado $\tilde{\mathcal{C}}$ expresado en coordenadas polares como la lista de valores donde la derivada de I_p respecto a la dirección radial se maximiza para

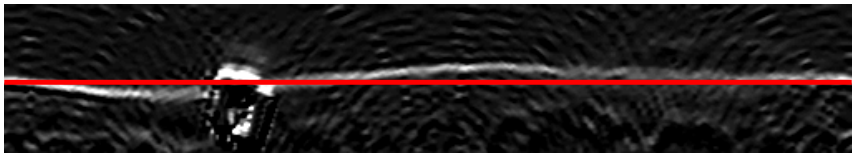


Figura 3.7: Derivada en la dirección y del aro en coordenadas polares. El contorno de la pupila aparece bien definido y cercano al contorno circular estimado en el paso anterior.

cada ángulo θ:

$$\tilde{\mathcal{C}}(\theta) = \arg\max_{\rho} \left| \frac{\partial I_p}{\partial \rho}(\theta, \rho) \right| \tag{3.11}$$

En la figura 3.8 se puede ver el resultado de calcular este máximo. El contorno entre la pupila y el iris está bien marcado, pero sin embargo existen errores generados al tener en cuenta la derivada en la región afectada por el reflejo del iluminador infrarrojo, y también existen algunos outliers generados por el ruido en el sistema de captura.

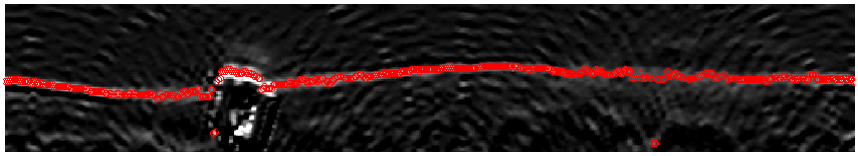


Figura 3.8: Resultado de calcular el máximo de la derivada en la dirección radial. El contorno de la pupila está bien marcado, excepto por la zona afectada por el iluminador y algunos outliers

Como se puede ver en dicha figura, el contorno resultante no es continuo y presenta varias imperfecciones. Como es deseable tener un contorno suave, una solución posible es suavizar esta contorno mediante un filtro gaussiano, pero esto no garantizaría que la curva sea continua.

Para garantizar la continuidad y hacer que el contorno sea suave al mismo tiempo, es posible analizar los coeficientes de la transformada de Fourier de $\tilde{\mathcal{C}}$, $\mathcal{F}\left\{\tilde{\mathcal{C}}\right\}$. Como en su mayor parte la curva es suave y continua (exceptuando los casos mencionados), cabe de esperar que con unos pocos coeficentes correspondientes a las frecuencias bajas de la señal, se pueda representar la forma de la curva sin ruido.

Asumiendo que el valor de θ está muestrado en k posiciones a intervalos regulares en el intervalo $[0 \ldots 2\pi)$, se tiene la transformada discreta de Fourier (DFT) de $\tilde{\mathcal{C}}$, $\mathcal{F}\left\{\tilde{\mathcal{C}}\right\}_i$, con $i \in \mathbb{Z}$, $-\frac{k}{2} < i < \frac{k}{2}$. Se define entonces la señal $\mathcal{F}_{(n)}\left\{\tilde{\mathcal{C}}\right\}_i$ de la siguiente manera:

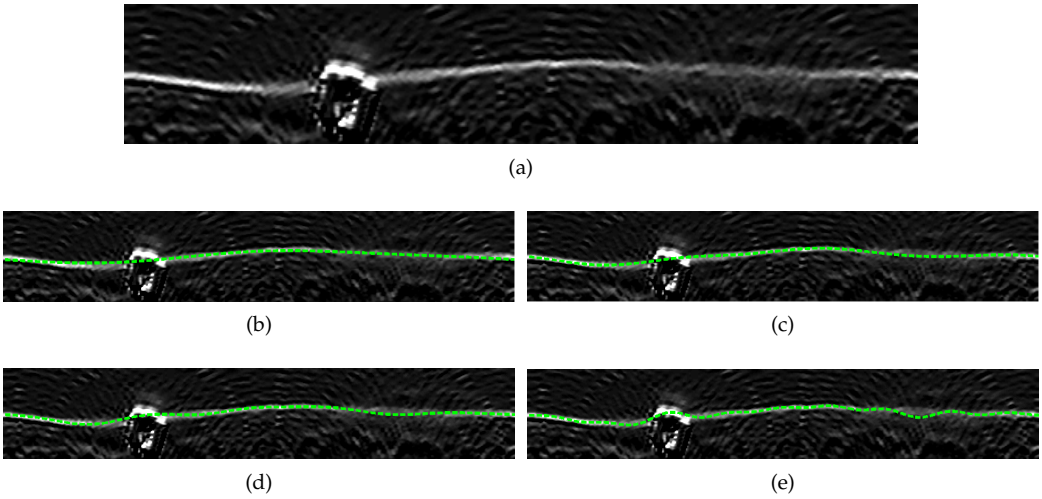
$$\mathcal{F}_{(n)}\left\{\tilde{\mathcal{C}}'\right\}_i = \begin{cases} \mathcal{F}\left\{\tilde{\mathcal{C}}\right\}_i & \text{Si } |i| \leq n \\ 0 & \text{Si } |i| > n \end{cases} \tag{3.12}$$

De esta forma, se eliminan los coeficientes correspondientes a las altas frecuencias de la señal. Finalmente, se define el contorno de la pupila expresado en coordenadas polares, $\mathcal{C}_{\text{polar}}$, como:

$$\mathcal{C}_{\text{polar}} = \mathcal{F}^{-1}\left\{\mathcal{F}_{(n)}\left\{\tilde{\mathcal{C}}\right\}\right\} \tag{3.13}$$

donde $\mathcal{F}^{-1}$ es la inversa de la transformada de Fourier.

En la figura 3.9 se puede ver el resultado de este método para una cantidad variable de coeficientes. Se puede ver con claridad que el método funciona muy bien aún en presencia de ruido y outliers, y que con muy pocos coeficientes se puede calcular el contorno con suma precisión.



(a) (b) (c) (d) (e)

Figura 3.9: Método de los coeficientes de Fourier para (3.9b) 2 coeficientes, (3.9c) 4 coeficientes, (3.9d) 6 coeficientes y (3.9e) 10 coeficientes. En verde, el mismo contorno que en la figura 3.8 pero suavizado. Se puede ver que para pocos coeficientes el método ajusta bien el contorno, pero para una cantidad mayor, el contorno empieza a ser afectado por el ruido

Una vez calculado el contorno expresado en coordenadas polares, es necesario convertir estos puntos a un contorno $\mathcal{C}_{\text{pupila}}$ expresado en coordenadas de la imagen:

$$\mathcal{C}_{\text{pupila}}(\theta) = \Big(x_{\text{pupila}} + \cos\theta \cdot \mathcal{C}_{\text{polar}}(\theta), y_{\text{pupila}} + \sin\theta \cdot \mathcal{C}_{\text{polar}}(\theta)\Big) \tag{3.14}$$

3.2.3. Segmentación del iris

Una vez localizada la pupila, el iris es relativamente fácil de ubicar ya que es casi concéntrico al círculo de la pupila (aunque este último suele poseer una desviación hacia el lado de la nariz respecto al círculo del iris [Dau04]).

En principio, tendría sentido utilizar el operador integro-diferencial para detectar la ubicación de la circunferencia del iris. Sin embargo, existen dos problemas:

- El iris puede estar obstruido en su parte inferior y/o superior por los párpados (figura 3.10)
- El contraste entre el iris y la esclera es mucho menor que entre la pupila y el iris, por lo cual el operador integro-diferencial será mucho más sensible al ruido.

Para solucionar estos problemas, se diseñó un algoritmo de segmentación optimizado para funcionar en tiempo real. El algoritmo se basa, al igual que el algoritmo de ajuste de contorno de la pupila, en extraer un aro de la imagen y convertirlo a coordenadas polares (figura 3.11).

Haciendo un análisis empírico de las imágenes capturadas, se puede observar que la región del iris comprendida entre los 45^o y -45^o y del centro de la pupila y la región entre los 135^o y 225^o suele tener muy poca oclusión causada por los párpados. En base a esta observación, se puede buscar el contorno del círculo limitando la búsqueda a estas regiones, y luego, interpolar el resultado al resto del iris, que puede o no presentar oclusión por parte de los párpados o las pestañas (figura 3.12).

En la imagen en coordenadas polares, cada ángulo del aro es representado por una columna, por lo que estas dos regiones equivalen a dos rectángulos en la imagen polar. Es de esperar entonces que en ambas regiones el contorno entre el iris y la esclera aparezca con poco ruido, bien

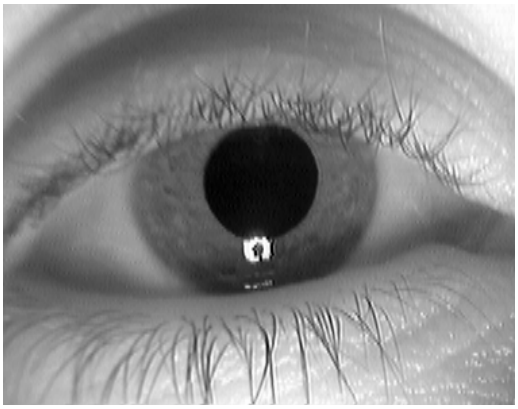
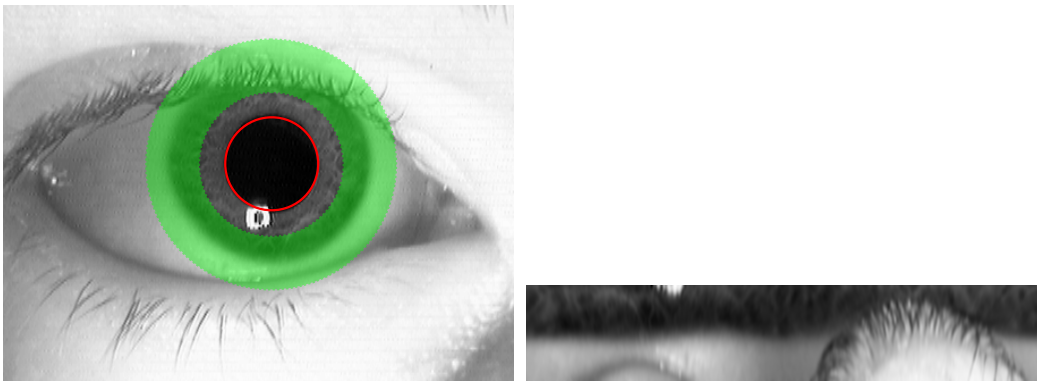


Figura 3.10: Imagen con el iris parcialmente ocluido por el párpado superior.



(a) (b)

Figura 3.11: Conversión a coordenadas polares de una corona circular que rodea la pupila. Esta corona contiene el contorno del iris, y puede poseer ruido de las pestañas y los párpados.

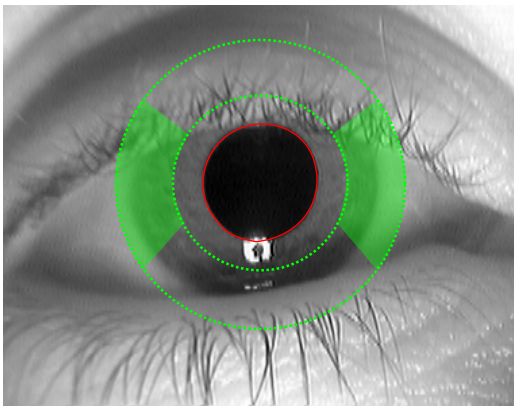
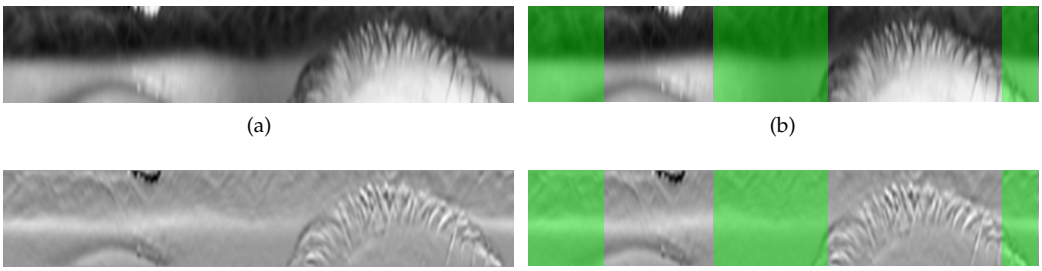


Figura 3.12: Zona de interés para el algoritmo de segmentación. La mayor parte del borde entre el iris y la esclera entran en esta zona.

definido y tenga una forma más o menos recta, dependiendo de la concentricidad de la pupila dentro del iris (figura 3.13).

Se busca entonces, en ambas zonas correspondiente a estas dos regiones en la imagen polar, el



(a) (b)

(c) (d)

Figura 3.13: Aro de la imagen que contiene al borde de iris en coordenadas polares (3.13a) y su derivada (3.13c). En verde se puede ver la región de interés para el algoritmo de segmentación (son dos regiones, sólo que la primera está cortada y aparece al principio y al final de la imagen)

radio (es decir, la fila) donde la derivada de dicha imagen sea máxima:

$$\rho_{\text{máx}_1} = \arg\max_{\rho} \int_{-\frac{\pi}{4}}^{\frac{\pi}{4}} \left| \frac{\partial I_p}{\partial r}(\theta, \rho) \right| \, \mathrm{d}\theta \tag{3.15}$$

$$\rho_{\text{máx}_2} = \arg\max_{\rho} \int_{\frac{3}{4}\pi}^{\frac{5}{4}\pi} \left| \frac{\partial I_p}{\partial r}(\theta, \rho) \right| \, \mathrm{d}\theta \tag{3.16}$$

En la figura 3.14 se ilustra el significado de estos dos valores.

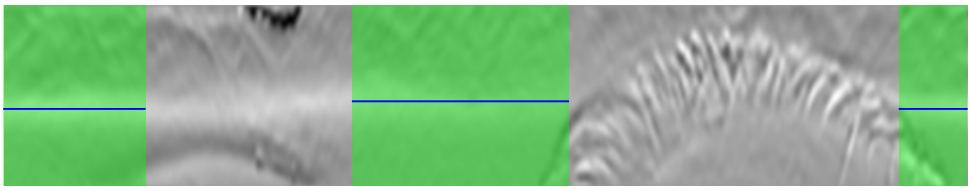


Figura 3.14: Aproximación del contorno dentro de las regiones de interés (verde). La línea azul representa el radio donde el valor de la derivada es máximo

Si la pupila está perfectamente centrada en el iris, se producirá que $\rho_{\text{máx}_1} = \rho_{\text{máx}_2}$, sin embargo, este no suele ser el caso. Por consiguiente, se realiza una interpolación lineal en el espacio de coordenadas polares entre los extremos de cada región para completar el círculo del iris. Se define entonces el el contorno $\mathcal{C}(\theta)$ de la siguiente forma:

$$\mathcal{C}(\theta) = \begin{cases} \rho_{\text{máx}_1} & \text{Si } \theta \in [-\frac{\pi}{4}, \frac{\pi}{4}] \\ \rho_{\text{máx}_1} + \frac{\theta - \frac{\pi}{4}}{\frac{3}{4}\pi - \frac{\pi}{4}} \left(\rho_{\text{máx}_2} - \rho_{\text{máx}_1} \right) & \text{Si } \theta \in (\frac{\pi}{4}, \frac{3}{4}\pi) \\ \rho_{\text{máx}_2} & \text{Si } \theta \in [\frac{3}{4}\pi, \frac{5}{4}\pi] \\ \rho_{\text{máx}_2} + \frac{\theta - \frac{5}{4}\pi}{\frac{7}{4}\pi - \frac{5}{4}\pi} \left(\rho_{\text{máx}_1} - \rho_{\text{máx}_2} \right) & \text{Si } \theta \in (\frac{5}{4}\pi, \frac{7}{4}\pi) \end{cases}$$

En la figura 3.15 se puede ver el resultado de aplicar esta interpolación en el contorno del iris.

Como la aproximación mediante la interpolación lineal no siempre da como resultado el contorno exacto del iris, se procede de una manera similar que con la pupila para encontrar el contorno exacto: para empezar, se permite que $\mathcal{C}$ varíe un poco para ajustarse al máximo de la deriva-

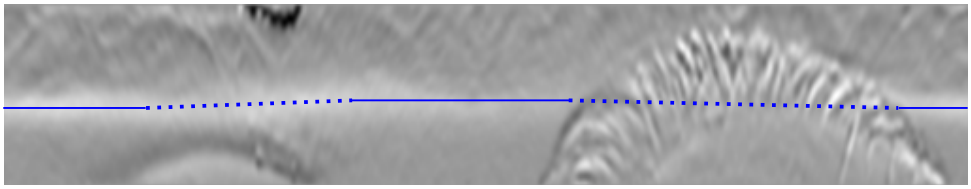


Figura 3.15: Interpolación de los valores obtenidos dentro de las regiones de interés a todo el contorno. La línea sólida representa el valor obtenido en el paso anterior, mientras que la línea punteada es la interpolación entre los extremos de las dos líneas.

da de I_p en la dirección radial:[2]

$$\tilde{\mathcal{C}}(\theta) = \arg\max_{\rho} \left\{ \left| \frac{\partial I_p}{\partial \rho}(\theta, \rho) \right| \ \forall \rho \in [\mathcal{C}(\theta) - \Delta\rho, \mathcal{C}(\theta) + \Delta\rho] \right\}$$

Luego, para eliminar el efecto del ruido y outliers, se suaviza la curva mediante el método de los coeficientes de Fourier, obteniendo el contorno final del iris:

$$\mathcal{C}_{\text{iris}} = \mathcal{F}^{-1}\left\{\mathcal{F}_n\left\{\tilde{\mathcal{C}}\right\}\right\}$$

Este método es sumamente robusto y permite encontrar el contorno del iris aún cuando hay una gran oclusión por parte de los párpados y pestañas.

3.2.4. Detección de párpados

Una vez localizados el iris y la pupila, es necesario ver qué parte del iris está ocluída por los párpados, a fin de excluir esta parte en el proceso de reconocimiento.

Es posible modelar los párpados como dos parábolas. La fórmula general para la parábola es la siguiente:

$$((y - y_0)\cos\theta - (x - x_0)\sin\theta)^2 = p\left((x - x_0)\cos\theta + (y - y_0)\sin\theta\right) \tag{3.17}$$

donde

- x_0 e y_0 representan el vértice de la parábola,
- p representa la apertura de la parábola, y
- θ representa la inclinación del eje la parábola respecto a los ejes

Al poseer cuatro parámetros, es computacionalmente muy costoso encontrar la posición de una parábola en la imagen en tiempo real utilizando la transformada de Hough o algún método similar (y, en este caso, es necesario encontrar dos parábolas: una para el párpado superior y otra para el párpado inferior).

Es necesario entonces acotar el rango de búsqueda. Una simplificación es asumir que $\theta = \frac{\pi}{2}$, es decir, que el eje de la parábola es perpendicular al eje de las abscisas. Esto ocurre cuando el ojo aparece en forma horizontal en la imagen. Bajo este supuesto, la fórmula queda:

$$(-(x - x_0))^2 = p(y - y_0) \tag{3.18}$$

luego

$$x^2 - 2xx_0 + x_0^2 = p(y - y0) \tag{3.19}$$

[2]Mientras que en la pupila se buscaba el máximo de la derivada en toda la columna, en este caso se busca el máximo de la derivada en una región más pequeña ya que el iris suele presentar mucho más ruido de las pestañas y párpados, y esto afecta los máximos de la derivada.

y, finalmente

$$y = \frac{1}{p}x^2 + \frac{-2x_0}{p}x + \frac{x_0^2 + py_0}{p} \tag{3.20}$$

con lo que se llega a la ecuación de la parábola de la forma $y = ax^2 + bx + c$. La ventaja de expresar la parábola en función de x_0, y_0 y p es que permite diseñar un algoritmo de segmentación utilizando el algoritmo integro-diferencial descripto anteriormente. De esta manera, sólo hace falta encontrar el punto (x_0, y_0) de la imagen donde haya un cambio brusco en el promedio de tonos de grises a lo largo de la parábola de apertura p en dicho punto, respecto a los puntos adyacentes.

Para encontrar estos tres parámetros, es sencillo adaptar el operador integro-diferencial para calcular el promedio a lo largo de la parábola en vez de a lo largo de una circunferencia. De hecho, es posible simplificar aún más el problema teniendo en cuenta que la forma de los párpados tiene muy poca variación (la curvatura es prácticamente constante), por lo que se puede asumir un valor de p constante para el párpado superior, p_{sup} y otro para el párpado inferior, p_{inf}, donde se debe verificar que $p_{\text{sup}} > 0$ y $p_{\text{inf}} < 0$ (ya que, en coordenadas de la imagen, el eje de las ordenadas es creciente "hacia abajo").

De esta forma, se construye una matriz M_p:

$$M_p(x_0, y_0) = \frac{1}{W_I} \sum_{x=0}^{W_I - 1} I(x, y) \tag{3.21}$$

donde y se saca de la ecuación 3.20 y $x_0 \in [0 \dots W_I]$ e $y_0 \in [0 \dots H_I]$, y W_I y H_I son el ancho y el alto de la imagen respectivamente. Los parámetros de la parábola están dados entonces por la ubicación en M donde se produce el máximo cambio en los tonos de grises, es decir, donde el módulo del gradiente es máximo:

$$(x_0, y_0) = \arg\max_{(x_0, y_0)} \left| \nabla M_p(x_0, y_0) \right| \tag{3.22}$$

La figura 3.16 muestra imágenes típicas obtenida por el sistema de captura en las cuales se superpusieron los resultados de los algoritmos de segmentación de la pupila, iris y párpados.

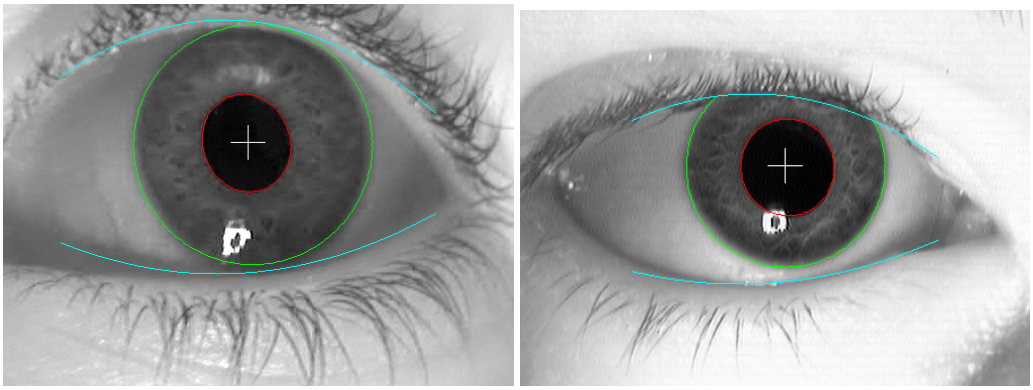


Figura 3.16: Resultados de los algoritmos de segmentación aplicados a imágenes típicas obtenidas por el sistema de captura.

En el capítulo 7 se analizarán los resultados y los tiempos de ejecución de los algoritmos de segmentación y se compararán los mismos con los resultados obtenidos en otros trabajos.

3.3. Resumen

En este capítulo se analizaron los problemas encontrados a la hora de segmentar las imágenes. Se propusieron tres algoritmos de segmentación de pupila, iris y párpados especialmente diseñados para funcionar en tiempo real. Los algoritmos propuestos mejoran el modelo circular clásico del iris y utilizan contornos más flexibles, y poseen la ventaja adicional de necesitar muy pocos parámetros para funcionar correctamente.

Capítulo 4

Análisis del video

En este capítulo se analiza el problema de capturar una imagen lo *suficientemente buena* del iris de una secuencia de video para su posterior procesamiento e identificación. La captura debe ser totalmente automática, y debe reconocer cuándo se presenta un cuadro (frame) que contiene una imagen de buena calidad de un iris.

Por lo general se realiza una comprobación acerca de la calidad de las imágenes como paso previo a la segmentación. En este trabajo, como se verá más adelante, se utiliza la información obtenida en la segmentación para establecer la calidad de las imágenes.

El sistema de captura se encarga de generar una secuencia de imágenes (*frames*) que debe ser procesada con el fin de encontrar una ”buena” imagen que permita el reconocimiento. El problema consiste entonces en definir qué condiciones debe tener una imagen para ser *buena*. En principio, se pueden definir las siguientes condiciones:

- Debe contener el iris de la persona a identificar.
- El iris debe estar (preferentemente) contenido en su totalidad dentro de la imagen.
- La región correspondiente al iris debe estar en foco, y no debe presentar *motion blur*.
- Dicha región debe contener suficiente información que permita la identificación. Para esto es necesario que se cumplan ciertas condiciones respecto al tamaño del iris en la imagen.

Cada una de estas condiciones son necesarias para la identificación, pero ninguna es suficiente por sí sola. Es necesario verificar cada condición por separado.

La manera óptima de comprobar que se cumplan estas condiciones es *en cascada*, es decir, comenzando por las condiciones más fáciles de comprobar, y, si alguna falla, finalizar el proceso asumiendo que la imagen no es buena. Recién una vez que todas las condiciones se cumplen se asume que la imagen es lo suficientemente buena.

No existe un único criterio que permita definir el orden óptimo en que se realizan las comprobaciones. Por ejemplo, en varios trabajos se comienza verificando el foco de la imagen ([HCTW06, Dau04]) y, si la misma no está en foco, se descarta automáticamente.

4.1. Detección de iris

Probablemente la condición más importante a satisfacer es que la imagen a procesar realmente contenga el iris de la persona a identificar. Si bien definir esto es trivial, es necesario implementar alguna técnica que permita decidir si en la imagen hay o no un iris.

La solución más común ante un problema de este tipo (reconocimiento de objetos) suele estar dado por algoritmos de clasificación basados en extracción de *features* de la imagen, como pueden ser los métodos basados en AdaBoost. Sin embargo, estos métodos presentan dos características no deseables:

- Son computacionalmente caros. Si bien algunos métodos están diseñados para funcionar en tiempo real (como el propuesto por Viola y Jones [VJ01]), sólo pueden procesar una cantidad bastante limitada de frames por segundo.
- Necesitan *entrenamiento*, es decir, es necesario presentarle al clasificador varias muestras del objeto a reconocer.

Analizando el problema particular de detectar la presencia o no de un iris en la imagen, es posible contar con el resultado obtenido en el proceso de segmentación. Este resultado permite obtener información específica sobre la localización y algunas características del iris en la imagen, si es que está presente.

Una observación importante es que el algoritmo de segmentación *siempre* da un resultado, independientemente si hay un iris o no en la imagen. Si no hay un iris, el resultado será aleatorio, mientras que si lo hay existe una gran probabilidad de que el resultado de la segmentación sea el correcto. Esto ocurre porque en ningún momento se está forzando el algoritmo de segmentación para que detecte un iris, sino que sólo se limita a, por ejemplo, encontrar la zona "más oscura" y asumir que ésa se corresponde con la pupila.

Luego, se puede analizar el resultado de la segmentación y, a partir de ahí, deducir de las características espaciales de la imagen si en la misma se encuentra presente un iris o no.

El resultado de la segmentación está dado por las circunferencias aproximadas del iris y de la pupila y por sus contornos exactos. Estos contornos cerrados definen en la imagen dos regiones, Ω_{pupila} y Ω_{iris} correspondientes a la pupila y al iris respectivamente. Como un punto no puede pertenecer a la pupila y al iris simultáneamente, se cumple que $\Omega_{\text{pupila}} \cap \Omega_{\text{iris}} = \emptyset$.

Luego, se definen las siguientes características de la imagen:

- Valor medio de gris de la pupila:

$$\mu_{\text{pupila}} = \frac{1}{n_{\text{pupila}}} \sum_{(x,y)\in\Omega_{\text{pupila}}} I(x,y) \tag{4.1}$$

- Valor medio de gris del iris:

$$\mu_{\text{iris}} = \frac{1}{n_{\text{iris}}} \sum_{(x,y)\in\Omega_{\text{iris}}} I(x,y) \tag{4.2}$$

- Varianza de los tonos de grises de la pupila:

$$\sigma^2_{\text{iris}} = \frac{1}{n_{\text{pupila}}} \sum_{(x,y)\in\Omega_{\text{pupila}}} \Big(I(x,y) - \mu_{\text{pupila}}\Big)^2 \tag{4.3}$$

- Varianza de los tonos de grises del iris:

$$\sigma^2_{\text{iris}} = \frac{1}{n_{\text{iris}}} \sum_{(x,y)\in\Omega_{\text{iris}}} \big(I(x,y) - \mu_{\text{iris}}\big)^2 \tag{4.4}$$

Con estos cuatro valores es posible armar una heurística que permita decidir si en la imagen se encuentra presente un iris o no:

1. Primero, se verifica que la región de la pupila sea mucho más oscura que la región del iris, es decir, que $\mu_{\text{iris}}/\mu_{\text{pupila}} > \kappa$, donde $\kappa > 1$ es un valor que define el contraste mínimo deseado entre las dos regiones. En la práctica, un valor de $\kappa = 1{,}5$ resulta satisfactorio.

2. Luego, se verifica que ambas regiones tengan tonos de gris uniformes. Para esto se verifica que $\sigma^2_{\text{pupila}} < k_{\text{pupila}}$ y que $\sigma^2_{\text{iris}} < k_{\text{iris}}$, donde k_{pupila} y k_{iris} son dos constantes que definen la flexibilidad en la varianza. Esta verificación tiene sentido ya que, si en la imagen no se encuentra presente un iris, es de esperar que la región que se detectó como iris contenga otros elementos que harán que la varianza no sea pequeña.

Comparar la varianza de la región del iris también tiene otra ventaja: permite descartar con facilidad las imágenes con alta oclusión por párpados y/o pestañas, ya que los mismos presentan un tono de gris distinto al tono de gris general del iris, y esto provoca que la varianza también aumente.

Estas dos heurísticas permiten entonces verificar que en la imagen exista una zona oscura y uniforme (la pupila) rodeada de otra más clara y también uniforme (el iris). Si bien las mismas pueden dar cierto número de falsos positivos (es decir, puede detectar iris en alguna imagen donde no los hay), al agregar una comprobación extra sobre la calidad de la imagen, como se verá a continuación, la cantidad de falsos positivos pasa a ser prácticamente nula.

4.2. Verificando la calidad de la imagen

En los trabajos existentes la principal métrica de calidad está relacionada con el nivel general de enfoque de la imagen. En [HCTW06] se define la *potencia de las altas frecuencias* de la imagen, o PHF por sus siglas en inglés *power of high frequencies*. La PHF de una imagen I está definido por:

$$\text{PHF}(I) = \iint_{\Omega} |\mathcal{F}\{I\}(u,v)|^2 \,\mathrm{d}u\,\mathrm{d}v \tag{4.5}$$

donde $\Omega = \{(u,v) : u^2 + v^2 > r^2\}$ y $\mathcal{F}\{I\}$ es la transformada de Fourier de I. Esencialmente lo que mide este operador es el aporte de las altas frecuencias en la imagen, ya que es de esperar que cuando una imagen está bien enfocada, los valores de las altas frecuencias son altos.

El problema con este método es que requiere calcular la transformada de Fourier de cada imagen. En una imagen de $n \times n$, esta operación toma $\mathcal{O}(n^2 \log n)$ operaciones, lo que es bastante costoso. Daugman [Dau04], en cambio, propone verificar el foco convolucionando la imagen con un kernel de 8×8:

$$K_{8\times 8} = \begin{array}{|cccccccc|}\hline -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & +3 & +3 & +3 & +3 & -1 & -1 \\ -1 & -1 & +3 & +3 & +3 & +3 & -1 & -1 \\ -1 & -1 & +3 & +3 & +3 & +3 & -1 & -1 \\ -1 & -1 & +3 & +3 & +3 & +3 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 & -1 & -1 & -1 \\ \hline \end{array} \tag{4.6}$$

Debido a que una convolución con el kernel en el dominio espacial equivale a una multiplicación en el dominio de las frecuencias, esta operación es similar a la realizada para calcular la PHF de la imagen, sólo que la convolución puede ser "optimizada", por ejemplo, calculando la convolución sólo en algunos puntos de la imagen. De esta forma la cantidad de operaciones se reduce de varios millones a unos pocos miles, lo que es mucho más adecuado en un entorno en tiempo real.

Como este kernel es la superposición de dos funciones rectangulares $\sqcap(x, y)$ (una de 8×8 y amplitud -1 y otra de 4×4 y amplitud 4) y la transformada de Fourier de la función rectangular es la función $\text{sinc}(u, v) = \frac{\sin(u)\sin(v)}{uv}$, la transformada de Fourier del kernel $K_{8\times8}$ tiene la fórmula:

$$\mathcal{F}\{K_{8\times8}\}(u, v) = \frac{\sin(u)\sin(v)}{\pi^2\, u\, v} - \frac{\sin(2u)\sin(2v)}{4\pi^2\, u\, v} \tag{4.7}$$

Este filtro, al igual que el operador PHF, deja pasar las altas frecuencias. Kang [KP07] propone un kernel similar al de Daugman, pero de 5×5:

$$K_{5\times5} = \begin{array}{|ccccc|}\hline -1 & -1 & -1 & -1 & -1 \\ -1 & -1 & +4 & -1 & -1 \\ -1 & +4 & +4 & +4 & -1 \\ -1 & -1 & +4 & -1 & -1 \\ -1 & -1 & -1 & -1 & -1 \\ \hline\end{array} \tag{4.8}$$

Las ventajas de este kernel es que al ser más pequeño requiere menor cantidad de operaciones, y tiene la característica de capturar mejor los valores de las altas frecuencias que el filtro de 8×8.

Todos los métodos descriptos tienen un problema significativo: son métodos globales que no tienen en cuenta las características de la imagen que se desea obtener. Los métodos establecen qué tan en foco está una *imagen* dada en vez de qué tan enfocado está el *iris* en la imagen (si es que éste existe).

Existen dos problemas comunes en los cuales éstos métodos fallan:

1. Imágenes donde las pestañas están enfocadas pero el iris está desenfocado.
2. Imágenes donde el usuario tiene lentes, y la imagen está enfocada en el lente (que suele contener polvo, rayas, etc.) en vez del iris.

De estos problemas, el primero es el más común y está directamente relacionado con la suposición de que la imagen enfocada presenta altas frecuencias. Las pestañas aparecen en la imagen como líneas muy contrastadas, y éstas aparecen como componentes de muy alta frecuencia. En la figura 4.1 se puede ver un ejemplo de esto.

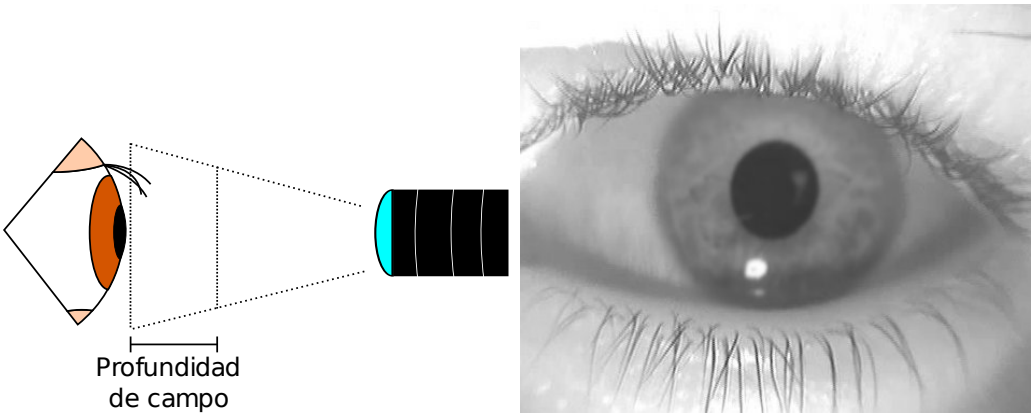


Figura 4.1: Ejemplo de captura donde las pestañas están enfocadas pero el iris no. La corta profundidad de campo del sistema de captura hace que sea posible tener las pestañas en foco sin enfocar el iris. Las pestañas aparecen en el dominio de las frecuencias como componentes de alta frecuencia.

4.2.1. Solución propuesta

Para solucionar estos dos problemas, se propone un método que contempla las características del iris. El mismo se basa en medir la calidad del contorno que separa la pupila del iris. La idea de este método es que cuando el iris está bien enfocado, el contorno entre la pupila y el iris está bien definido.

Como se vio en el capítulo 3, el contorno entre la pupila y el iris está dado por una función $\mathcal{C}_{\text{pupila}} : [0 \dots 2\pi) \to \mathbb{R}^2$, que a su vez está calculada a partir de la representación del contorno en coordenadas polares, $\mathcal{C}_{\text{polar}}$ (ecuación 3.14).

Volviendo a la idea de la imagen en coordenadas polares, I_p, el contorno es una curva unidimensional que separa la región de "arriba" (la pupila) de la de "abajo" (el iris). Cabe esperar que mientras mejor enfocada esté la imagen, este borde será más nítido, y por lo tanto concentrará una mayor parte de la energía de la derivada en la dirección radial a lo largo de este contorno.

De esta forma, se define la energía total de la derivada de I_p en la dirección radial de la siguiente manera:

$$E_{\text{total}} = \iint \left| \frac{\partial I_p}{\partial \rho}(\theta, \rho) \right|^2 \mathrm{d}\rho \, \mathrm{d}\theta \tag{4.9}$$

La energía contenida en el borde de las dos regiones se define como:

$$E_{\text{borde}} = \int_0^{2\pi} \int_{\mathcal{C}_{\text{pupila}}(\theta)-\Delta\rho}^{\mathcal{C}_{\text{pupila}}(\theta)+\Delta\rho} \left| \frac{\partial I_p}{\partial \rho}(\theta, \rho) \right|^2 \mathrm{d}\rho \, \mathrm{d}\theta \tag{4.10}$$

El valor de $\Delta\rho$ permite establecer qué tan grande es la región alrededor del borde en la cual se está midiendo la energía. Se define entonces la calidad del borde, q_{borde}, como:

$$q_{\text{borde}} = 100 \frac{E_{\text{borde}}}{E_{\text{total}}} \tag{4.11}$$

con $0 \leq q_{\text{borde}} \leq 100$.

En las figuras 4.2 y 4.3 se puede ver este método aplicado a dos imágenes, una desenfocada y otra enfocada. Para la desenfocada resulta $q_{\text{borde}} = 32$ mientras que para la enfocada resulta $q_{\text{borde}} = 50$.

Finalmente se debe establecer un criterio para definir cuándo una imagen está lo suficientemente en foco. Experimentalmente se tiene que con $q_{\text{borde}} \geq 60$ la calidad de las imágenes son razonablemente buenas, por lo que este será el umbral mínimo $q_{\text{mín}}$ de calidad para las imágenes.

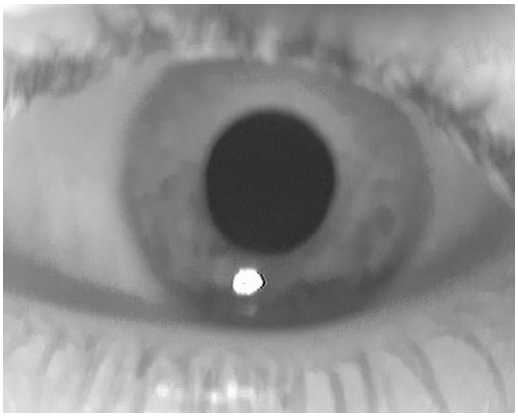
Una ventaja de este método es que se puede realizar en forma paralela con el proceso de segmentación, en el momento en que se detecta el contorno de la pupila en la corona en coordenadas polares.

4.3. Detección de la imagen óptima en la secuencia de video

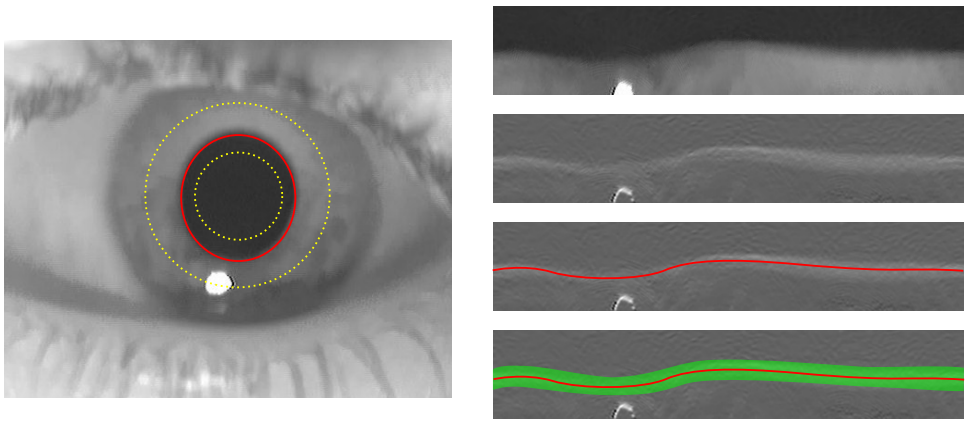
Hasta ahora se vio cómo detectar, en una secuencia de video, las imágenes que contienen un iris y aquellas que están bien enfocadas. Una ventaja de contar con una secuencia de video provista por el sistema de captura es que es posible no conformarse con *una* buena imagen, sino que se puede obtener la *mejor* imagen de la secuencia.

Usando como métrica de la calidad de la imagen la calidad del borde de la pupila, en la figura 4.4 se puede ver cómo evoluciona la calidad de la imagen cuando el usuario se acerca a la cámara. Se pueden distinguir cinco etapas (figura 4.4):

1. El usuario está demasiado lejos de la cámara. El algoritmo de segmentación no logra ubicar la pupila y el valor de enfoque es un valor casi aleatorio.



(a)



(b)

Figura 4.2: Cálculo de la calidad de la imagen para una imagen desenfocada. 4.2a: imagen capturada. 4.2b: extracción de una corona (en amarillo) y conversión a coordenadas polares. La primer imagen de la columna derecha corresponde a la corona en coordenadas polares, I_p. La segunda imagen es la derivada en la dirección radial, $\frac{\partial I_p}{\partial \rho}$. La tercer imagen es la derivada con el borde del iris delineado en rojo. La cuarta imagen contiene la región (en verde) en la cual se calcula la energía de la derivada para verificar la calidad de la imagen.

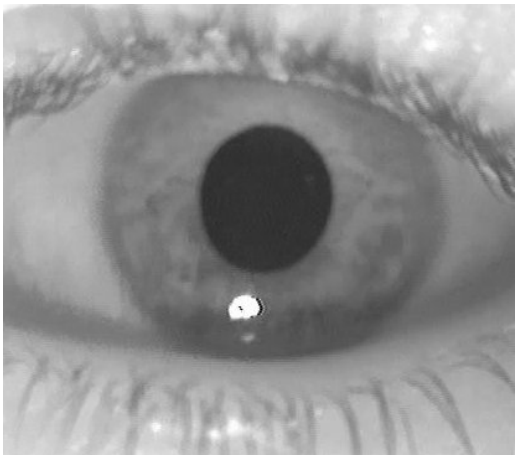
2. El usuario se acerca lo suficiente como para que el algoritmo de segmentación ubique la pupila. El valor de enfoque se empieza a estabilizar, pero está por debajo del valor aceptable de enfoque.

3. A medida que el usuario se acerca a la cámara y a la distancia de enfoque, el valor de enfoque empieza a crecer, hasta que llega a un máximo local cuando su distancia a la cámara es igual a la distancia de enfoque.

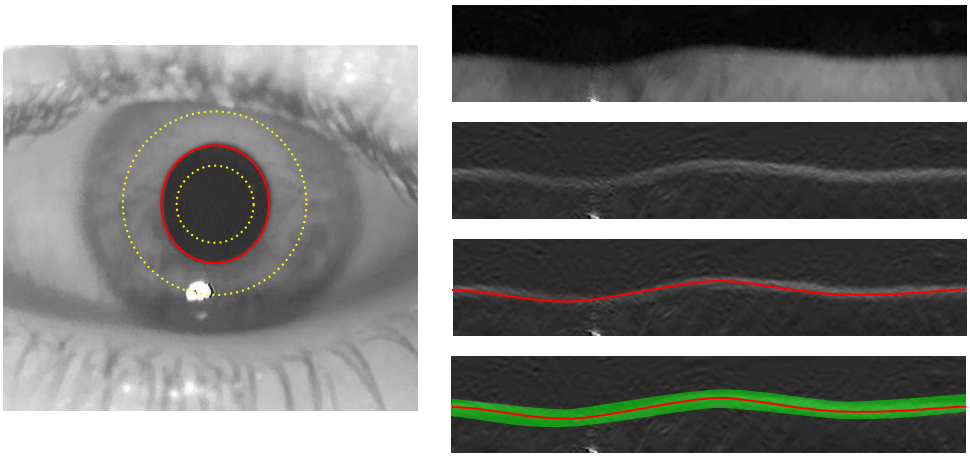
4. Una vez pasado este punto, el valor de enfoque comienza a decrecer nuevamente cuando el usuario se acerca a la cámara.

5. Cuando el usuario comienza a alejarse, pasa nuevamente por la distancia de enfoque, lo que se traduce en otro máximo local. Luego, al seguir alejándose, el valor de enfoque sigue decreciendo.

Es necesario entonces detectar cuándo se producen estos máximos locales en la calidad de la



(a)



(b)

Figura 4.3: Ídem figura 4.3 pero con una imagen mejor enfocada. Se puede ver que concentra una mayor cantidad de energía a lo largo del contorno entre el iris y la pupila (éste aparece más definido en la derivada)

imagen en tiempo real. Para esto, se trabaja con un buffer de tamaño fijo que almacena las últimas imagenes capturadas.

El buffer trabaja en forma de cola: cuando se captura una imagen nueva, se agrega la nueva imagen al final del buffer y la primer imagen del buffer se descarta. La idea es que cuando se alcanza el máximo local, en algún momento éste quedará al principio del buffer y podrá ser identificado como tal. El buffer hace de ventana cuyo tamaño define el rango en el cual se buscarán los máximos locales. El algoritmo 4.1 muestra esta operación.

4.4. Enfoque automático

Lo descripto hasta ahora asume que el sistema de captura utiliza un lente de foco fijo, como se explicó en el capítulo 2, y que el usuario era el encargado de acercarse al sistema de captura hasta ubicarse a la distancia adecuada.

Si el sistema de captura cuenta con un lente con foco regulable mediante software[1], se puede

[1]Un lente autofocus puede tener problemas de enfoque con las pestañas, como se describió anteriormente, así que se asume que el foco es ajustable por software programable.

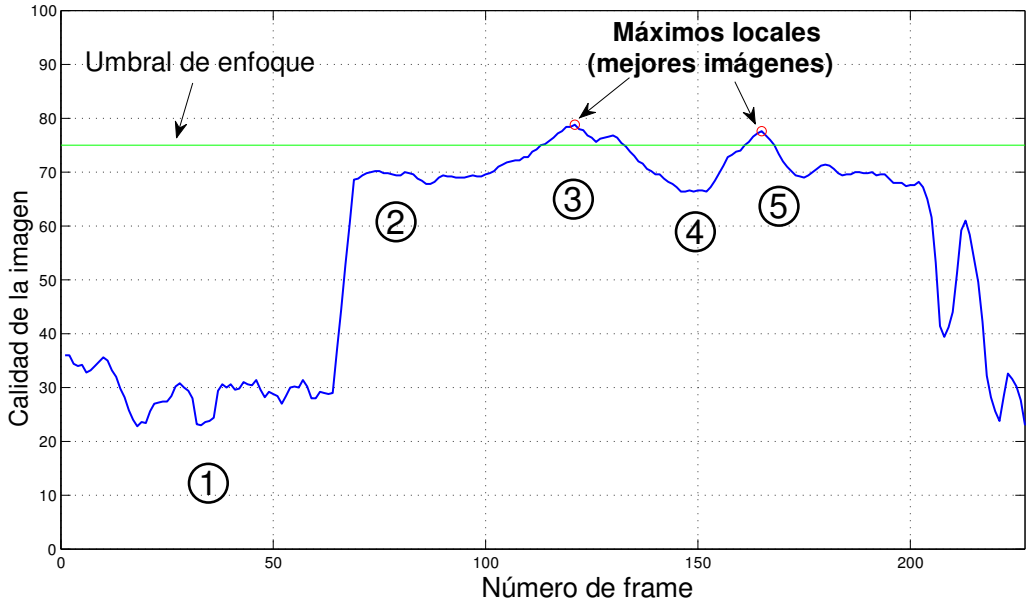


Figura 4.4: Evolución del valor de enfoque cuando el usuario se acerca a la cámara

Algoritmo 4.1: Detección de la imagen óptima

Sea B una cola de tamaño n

Mientras se capture una imagen I e I contenga un iris
  Encolar I en B
  Si calidad$(B_0) >$ calidad$(B_j)\ \forall j = 1 \ldots n-1$ y calidad$(B_0) \geq q_{\text{mín}}$
    Identificar B_0 como la mejor imagen
    Utilizar B_0 para la identificación
  Fin
Fin

simplificar el proceso desde el punto de vista del usuario.

De esta forma, el usuario puede posicionarse a una distancia *aproximada* del plano de enfoque y el sistema puede enfocar automáticamente en el iris del usuario. Para esto, se puede utilizar la misma métrica de calidad descripta anteriormente. La diferencia es ahora que, en vez de esperar el máximo local en forma *pasiva*, el sistema debe modificar el plano de enfoque del lente buscando dicho máximo en forma *activa*.

En el lente autofocus, se disponen de dos operaciones posibles: acercar el plano de enfoque y alejar el plano de enfoque. Con esto, es sencillo diseñar un algoritmo de enfoque automático a partir de las siguientes reglas:

1. Cuando el sistema detecta un iris, calcular la calidad de la imagen y alejar (o acercar) el plano de enfoque levemente.

2. Capturar algunas imágenes. Si las mismas tienen una calidad *menor*, quiere decir que el plano de enfoque fue movido hacia el lado equivocado, por lo que se invierte el movimiento del lente, enfocando en la dirección contraria. De lo contario, si la calidad es *mayor*, significa que el plano de enfoque está siendo movido en la dirección correcta, por lo que se debe seguir enfocando en ese sentido.

3. Seguir modificando el plano de enfoque hasta encontrar un máximo local, con el algoritmo 4.1.

4.5. Resumen

En este capítulo se analizó el problema de capturar ”buenas” imágenes de forma automática. El objetivo de esto es detectar el momento en que el usuario se posiciona a la distancia adecuada del sistema de captura y se obtiene una imagen del iris de la persona con suficiente calidad para la identificación.

Este paso es de suma importancia en el proceso de reconocimiento, ya que la performance del sistema depende directamente de la calidad de las imágenes obtenidas. Las imágenes de baja calidad (fuera de foco, movidas, etc) pueden dar lugar a errores en la etapa de reconocimiento, como falsos rechazos o falsas aceptaciones. Esto se verá con más detalle en el capítulo 5.

Capítulo 5

Codificación y matching

La **codificación** del iris es el proceso por el cual se analiza la textura del iris y se extrae información, mientras que el *matching* es la comparación de la información extraída con el fin de tomar una decisión acerca de la identidad del usuario. En este capítulo se hará un repaso de algunos de los métodos existentes y se describirá el método implementado en el sistema desarrollado.

5.1. Métodos de codificación y matching existentes

La codificación debe ser capaz de extraer aquella información del iris que permita discriminar una persona de otra. El resultado de la codificación es por lo general un vector de características o *features* que describen la textura del iris. Dicho vector debe poder ser comparado contra otros vectores generados a partir de otros iris o del mismo iris. En la literatura existen muchos métodos propuestos para extraer características del iris, y cada uno de estos métodos tiene asociado un método de matching.

La codificación del iris es probablemente el área que más se ha estudiado y sobre el que existen mayor cantidad de trabajos. Entre todos los métodos existentes, los más comunes y aquellos en los que se han obtenido mejores resultados son los que utilizan filtros de Gabor o Log-Gabor y los que utilizan wavelets para codificar la textura del iris. Se hará a continuación un repaso de los métodos más utilizados y los más interesantes desde el punto de vista de la originalidad y eficacia.

5.1.1. Filtros de Gabor

En su sistema original, Daugman [Dau93] utilizó los filtros de Gabor bidimensionales como método de codificación del iris. Los filtros de Gabor son un tipo de filtro pasabanda (filtro que deja pasar un determinado rango de frecuencias de una señal) que determina el rango de frecuencias utilizando una función Gaussiana [JHG99]. Estos filtros surgen del análisis del funcionamiento de las neuronas en la corteza visual primaria [Dau80]. Dichas neuronas reaccionan detectando la *orientación* de los objetos.

En el domino de las frecuencias, el filtro de Gabor $\hat{G}(u,v)$ se define como una función Gaussiana:

$$\hat{G}(u,v) = \exp\left[-\pi\left((u-u_0)^2\alpha^2 + (v-v_0)^2\beta^2\right)\right] \tag{5.1}$$

donde (u_0, v_0) representa la modulación (es decir, el punto donde la respuesta del filtro es máxima) y α y β representan la desviación de la función Gaussiana bidimensional (el cociente α/β define la relación de aspecto del filtro). En el dominio espacial, el filtro de Gabor centrado en el

origen está dado por:

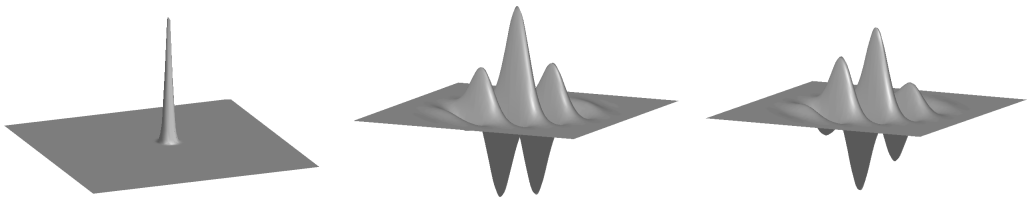
$$G(x,y) = \mathcal{F}^{-1}\left\{\hat{G}\right\} = \exp\left(-\pi\left(\frac{x^2}{\alpha^2} + \frac{y^2}{\beta^2}\right)\right)\exp\left(-2\pi i\left(u_0 x + v_0 y\right)\right) \tag{5.2}$$

El filtro resultante es entonces una onda sinusoide compleja (llamada *carrier*) modulada por una función Gaussiana (llamada *envelope*). La frecuencia espacial del filtro es $\omega_0 = \sqrt{u_0^2 + v_0^2}$ y su orientación es $\theta_0 = \arctan(v_0/u_0)$, mientras que el ancho de banda está dado por el radio de la Gaussiana (definido por α y β).

Este filtro se puede separar en dos componentes, una real y una compleja:

$$\begin{aligned} G_{\text{Re}}(x,y) &= g_{\alpha,\beta}(x,y)\cdot\cos\left(2\pi(u_0 x + v_0 y)\right) \\ G_{\text{Im}}(x,y) &= g_{\alpha,\beta}(x,y)\cdot\sin\left(2\pi(u_0 x + v_0 y)\right) \end{aligned} \tag{5.3}$$

siendo $g_{\alpha,\beta}(x,y) = \exp\left(-\pi\left(\frac{x^2}{\alpha^2} + \frac{y^2}{\beta^2}\right)\right)$ el envelope Gaussiano. Este par de filtros están en *cuadratura*, lo que quiere decir que tienen la misma amplitud pero están desfasados 90^o. El filtro G_{Re} se denomina *filtro simétrico par* y G_{Im} se denomina *filtro simétrico impar*, ya que el coseno y el seno son funciones simétricas par e impar respectivamente. En la figura 5.1 se puede ver un ejemplo de las componentes de un filtro de Gabor.



(a) (b) (c)

Figura 5.1: Ejemplo de filtro de Gabor. La figura 5.1a muestra el filtro en el dominio de las frecuencias. Las figuras 5.1b y 5.1c muestran el filtro real y complejo en el dominio espacial, respectivamente.

La utilidad de este filtro radica en que está demostrado que minimiza el principio de incertidumbre en los dominios espacial y de frecuencias. Este principio establece la imposibilidad de tener información en el dominio de las frecuencias y en el dominio espacial *simultáneamente* [Dau88]. El filtro de Gabor permite tener información en ambos dominios minimizando la incertidumbre conjunta.

Generación del código

Para generar el código de iris, Daugman propone primero normalizar las coordenadas de los puntos del mismo, expresando cada punto como un par (θ, r) en coordenadas polares. Esto permite que el método sea invariante frente a cambios en el tamaño del iris, ya sea por la distancia del ojo a la cámara o por la contracción o dilatación de la pupila. La textura del iris queda entonces expresada como una función $I(\theta, r)$.

Luego, se proyecta un área del iris centrada en el punto (θ_0, r_0) en una wavelet bidimensional de Gabor[1]. Este procedimiento se repite para varios puntos y varias wavelets a distintas escalas y frecuencias.

[1]Una *wavelet* de Gabor es una familia de wavelets que utiliza la función de Gabor descripta como función madre e incluye translaciones y cambios de escala.

Cada aplicación de una wavelet en un punto da como resultado un número complejo $c = a + ib$ que puede ser expresado en función de su *argumento* o *fase*, $\arg(c)$ y su *amplitud*, $|c|$. Está demostrado que la fase y no la amplitud es la que provee la información más significativa en una imagen [OL81], por lo que la amplitud, que contiene información redundante, es ignorada en la codificación.

La fase, al ser un número real entre 0 y 2π es cuantizada a fin de reducir la complejidad del código resultante. Ésta se codifica con dos bits, que indican el cuadrante en el cual está ubicado el resultado. Este proceso se puede ver con la siguiente fórmula:

$$h_{\{\mathrm{Re,Im}\}} = \mathrm{sgn}_{\{\mathrm{Re,Im}\}} \int_\rho \int_\phi I(\phi,\rho) \exp\left(-i\omega(\theta_0 - \phi)\right) \cdot \\ \cdot \exp\left(-\left(\frac{(r_0-\rho)^2}{\alpha^2} + \frac{(\theta_0-\phi)^2}{\beta^2}\right)\right) \rho \, \mathrm{d}\phi \, \mathrm{d}\rho \quad (5.4)$$

El código resultante es el conjunto de bits obtenidos al aplicar este procedimiento a varios puntos con varias wavelets en distintas escalas y frecuencias, obteniendo un total de 2048 bits.

Además del código de iris, se genera una matriz binaria que marca aquellos puntos que fueron afectados por ruido proveniente de las pestañas o los párpados y que por lo tanto no pertenecen al iris. Esta matriz funcionará como máscara en la etapa de matching.

Matching

El matching consiste en comparar dos códigos de iris y decidir si ambos fueron generados por el mismo iris o no. En el sistema descripto por Daugman, cada código de iris es una matriz binaria que también tiene asociada una máscara de ruido que marca aquellos bits que fueron afectados por las pestañas y/o los párpados. Para comparar dos códigos, se calcula la *distancia de Hamming* entre ambos códigos, teniendo en cuenta sus máscaras:

$$HD(C_1, M_1, C_2, M_2) = \frac{\|(C_1 \otimes C_2) \cap (M_1 \cap M_2)\|}{\|M_1 \cap M_2\|} \quad (5.5)$$

donde $\otimes$ es el operador binario XOR y $\cap$ es el operador binario AND.

Esencialmente, la distancia de Hamming mide la cantidad de bits en ambos códigos que son desiguales entre sí. Este valor es normalizado por la cantidad de bits válidos en ambos códigos (definidos por la intersección entre ambas máscaras), lo que da como resultado un número entre 0 y 1. Mientras más parecidos sean los códigos de iris, la cantidad de bits distintos será cada vez menor y por lo tanto la distancia de Hamming será cercana a 0.

Esta métrica permitirá tomar una decisión acerca de la identidad de la persona. Esto se verá con más detalle en el capítulo 7.

5.1.2. Filtros de Log-Gabor

Un problema con los filtros de Gabor es que no se puede construir un filtro con un ancho de banda arbitrariamente grande y mantener al mismo tiempo una respuesta baja en el coeficiente DC del filtro simétrico par (el filtro simétrico impar no sufre este problema). Esto implica que a medida que la respuesta del filtro se acerca al coeficiente DC, la misma será afectada por cambios de iluminación. Los filtros con ancho de banda superior a una octava sufren de este problema.

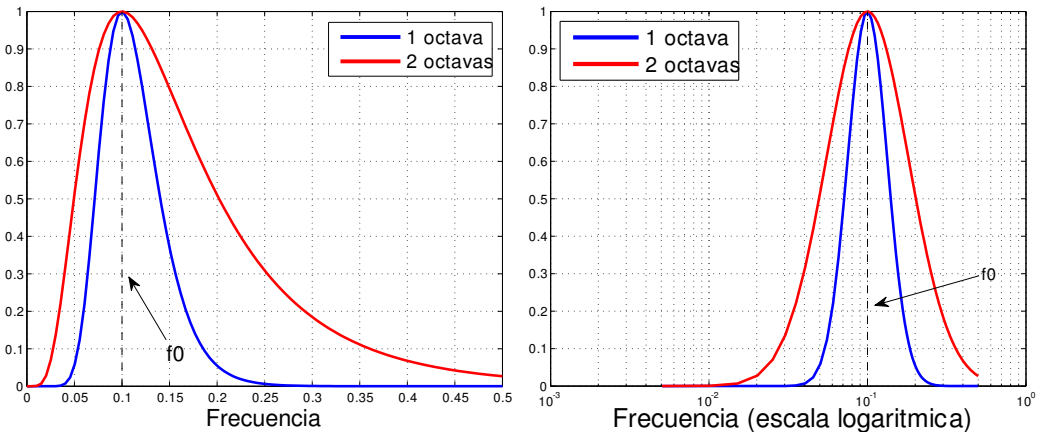
Field [Fie87] propone los filtros de Log-Gabor como alternativa a los filtros de Gabor para codificar imágenes naturales. Los filtros de Log-Gabor tienen, al igual que los filtros de Gabor, una función de transferencia Gaussiana pero en escala *logarítmica*. Según Field, esto codifica mejor las imágenes naturales que los filtros de Gabor.

La función de transferencia de un filtro de Log-Gabor unidimensional está definido en el espacio de las frecuencias como:

$$\hat{G}(f) = \exp\left(\frac{-\log(f/f_0)^2}{2\log(\sigma/f_0)^2}\right) \tag{5.6}$$

donde f_0 es la frecuencia central donde la respuesta del filtro es máxima, y σ define el ancho de banda del filtro. Estos filtros permiten un ancho de banda arbitrariamente grande, manteniendo siempre una respuesta nula en el coeficiente DC. El único problema con estos filtros es que no existe una fórmula cerrada para el filtro en el dominio espacial, por lo que debe ser calculado numéricamente a partir de la transformada inversa de Fourier de la función de transferencia $\hat{G}(f)$.

En las figuras 5.2 y 5.3 se pueden ver las funciones de transferencia de dos filtros con ancho de banda de una y dos octavas, y los filtros resultantes, respectivamente.



(a) (b)

Figura 5.2: Ejemplo de filtros de Log-Gabor en el dominio de las frecuencias en escala real y logarítmica. Ambos filtros tienen la misma frecuencia central $f_0 = 0{,}1$, y un cociente $\sigma/f_0 = 0{,}75$ para el filtro de una octava y $\sigma/f_0 = 0{,}55$ para el de dos octavas (valores obtenidos de [Kov99]). Se puede ver cómo el filtro tiene una función de transferencia Gaussiana en escala logarítmica.

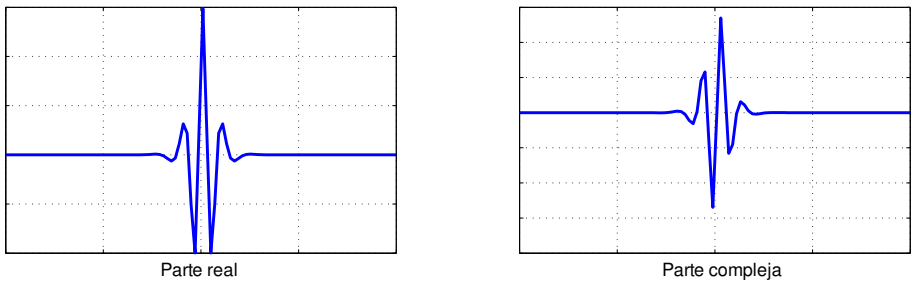
Generación del código

Los filtros de Log-Gabor fueron utilizados por Masek [Mas03] y por Huang [HMWT] para codificar texturas de iris. Masek utiliza el filtro unidimensional descripto para codificar "anillos" del iris: cada anillo es convolucionado con el filtro y el resultado es cuantizado de la misma forma descripta por Daugman, generando un código de iris de 9600 bits.

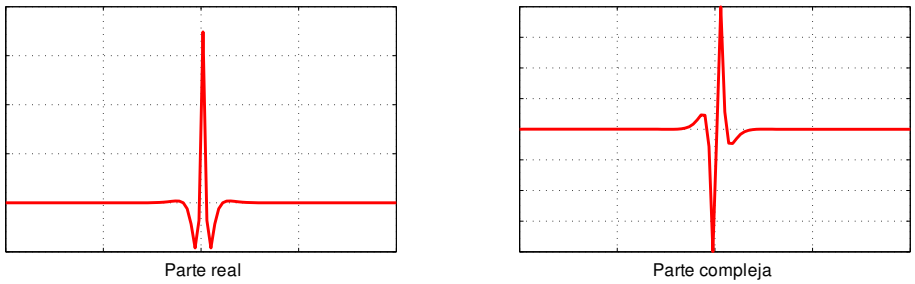
Huang, en cambio, genera un banco de filtros bidimensionales utilizando los filtros de Log-Gabor. Estos filtros se expresan en coordenadas polares en el espacio de las frecuencias como el producto de un filtro de Log-Gabor en la dirección radial y un envelope Gaussiano:

$$\hat{G}(\theta,\rho) = \exp\left(\frac{-\log(\rho/\rho_0)^2}{2\log(k/\rho_0)^2}\right)\exp\left(\frac{-(\theta-\theta_0)^2}{2T(\Delta\theta)^2}\right) \tag{5.7}$$

La textura del iris es convolucionada con seis filtros, cada uno con una orientación distinta (0^o, 30^o, 60^o, 90^o, 120^o y 150^o). Para cada punto del iris, se elige la orientación donde la respuesta del filtro sea máxima y con esto se construye el código del iris, que tiene un total de 1280 valores (a diferencia de Daugman y Masek, el código de iris es un conjunto de números reales en vez de bits). Como método de matching entre dos códigos, se utiliza simplemente la distancia euclídea entre los mismos.



(a)



(b)

Figura 5.3: Parte real y parte compleja de los filtros de Log-Gabor en el dominio espacial. Al igual que los filtros de Gabor, ambas partes están desfasadas 90^o, lo que genera un par de filtros en cuadratura.

5.1.3. Wavelets

La codificación mediante wavelets fue propuesta por Boles y Boashash [BB98]. Su sistema codifica anillos del iris (denominados *círculos virtuales*) utilizando la representación por cruces de ceros de la transformada wavelet diádica del anillo.

La representación por cruces de ceros de una señal f se define de la siguiente manera [Mal91]: si z_{n-1} y z_n son dos ceros consecutivos de la señal, es decir, si $f(z_{n-1}) = f(z_n) = 0$ y $f(x) \neq 0 \; \forall x \in (z_{n-1}, z_n)$, se define

$$e_n = \int_{z_{n-1}}^{z_n} f(x)\,\mathrm{d}x \tag{5.8}$$

y la representación por cruces de ceros de f, Zf, está dada por

$$Zf(x) = \frac{e_n}{z_n - z_{n-1}} \text{ para } x \in [z_{n-1} \ldots z_n] \tag{5.9}$$

Para comparar dos códigos f y g, define dos funciones de disimilaridad:

$$d_j^{(1)}(f,g) = \min_m \sum_{n=1}^{N} \left| Z_j f(n) - \Gamma \cdot Z_j g(n+m) \right|^2 \tag{5.10}$$

$$d_j^{(2)}(f,g) = \min_m \frac{\sum_{r=1}^{R_j} \left| \Phi_j^f(r) - \Gamma \cdot \Phi_j^g(r+m) \right|^2}{\Gamma \cdot \sum_{r=1}^{R_j} \left| \Phi_j^f(r) \right| \cdot \left| \Phi_j^g(r) \right|} \tag{5.11}$$

donde j representa el nivel de la wavelet, Z_j representa los cruces de ceros de la wavelet en el j-ésimo nivel, $\Phi_j^f(r)$ representa el producto entre la posición y la magnitud del r-ésimo cruce de ceros de $Z_j f$ y Γ es un factor que compensa la diferencia entre los radios del iris. La existencia de dos funciones se debe a que la segunda funcion requiere menos cálculos que la primera, pero sólo se puede utilizar cuando la cantidad de cruces de ceros en las dos señales es igual.

Ma [MTWZ04] también utiliza anillos del iris y la transformada wavelet para generar un código de iris. Sin embargo, en vez de utilizar los cruces de ceros para generar el código, codifica las posiciones de los máximos y mínimos locales consecutivos de la transformada wavelet en dos niveles. Para comparar dos códigos, genera una representación binaria de cada código y las compara utilizando la operación XOR normalizada.

Lim [LLBK01], en cambio, utiliza la wavelet bidimensional de Haar para realizar una descomposición en cuatro niveles, y genera un código sumamente compacto de 87 bits, utilizando una red neuronal en la etapa de matching y clasificación.

5.1.4. Laplaciano de Gaussianas

El método de codificación por Laplaciano de Gaussianas fue introducido por Wildes [WAG+94]. Consiste en generar representaciones de la imagen del iris en varias escalas. Cada escala es filtrada con un filtro cuasi-Gaussiano, definido por

$$w = -\frac{1}{\pi\sigma^4}\left(1 - \frac{\rho^2}{2\sigma^2}\right)\exp\left(-\frac{\rho^2}{2\sigma^2}\right) \tag{5.12}$$

y el filtro bidimensional aplicado en cada escala es $W = w^T w$, obteniendo un total de cuatro niveles. Cada nivel está definido por el nivel anterior como:

$$g_k = (W * g_{k-1})_{\downarrow 2} \tag{5.13}$$

con $g_0 = I$. Se define entonces la pirámide Laplaciana l_k como la diferencia entre dos niveles consecutivos, donde el tamaño del menor nivel es ajustado al de mayor nivel:

$$l_k = g_k - 4W * (g_{k+1})_{\uparrow 2} \tag{5.14}$$

La ventaja de esta representación es que se pueden comparar dos imágenes de iris simplemente buscando el máximo de la correlación normalizada entre las dos imágenes. Esto se hace para los cuatro niveles de la pirámide, obteniendo un total de cuatro valores con los cuales se puede realizar el matching.

5.1.5. Otros métodos

Correlación de fase

Miyazawa [MIA05] propone comparar dos imágenes normalizadas de iris f y g mediante la *correlación únicamente de fase* (en inglés, *phase-only correlation*). La correlación común se calcula como producto de las transformadas de Fourier de f y g, F y G respectivamente:

$$\mathrm{Corr}(f, g) = \mathcal{F}^{-1}\{F \cdot G\} \tag{5.15}$$

La correlación de fase utiliza únicamente la diferencia entre las fases de F y G:

$$\mathrm{Corr}_{\text{fase}}(f, g) = \mathcal{F}^{-1}\left\{\frac{F \cdot \bar{G}}{|F \cdot \bar{G}|}\right\} \tag{5.16}$$

donde $\bar{G}$ representa el conjugado complejo de G. La ventaja de este método es que, por un lado, al utilizar únicamente la fase es insensible frente a cambios de iluminación, y por otro lado es posible calcular la correlación sólo para un conjunto de frecuencias deseado, lo que minimiza los efectos del ruido (esto se denomina *band-limited phase-only correlation*).

DCT

Monro [MRZ07] utiliza la transformada discreta del coseno (DCT) para codificar "parches" de la textura del iris. Cada parche se codifica utilizando la representación por cruce de ceros y luego es cuantizado en un código binario. Se utiliza la distancia de Hamming para realizar el matching entre dos códigos de iris.

5.2. Implementación

En el sistema implementado, se eligió el método de codificación mediante filtros de Log-Gabor unidimensionales debido a los buenos resultados reportados por Masek [Mas03] y Terissi [TCB06] con este método y a la simplicidad del método. A continuación se hará una descripción detallada del método implementado.

5.2.1. Normalización

Previo a la codificación, se extrae la textura del iris en una nueva imagen rectangular de dimensiones fijas. Cada punto del iris puede ser referenciado, independientemente del tamaño, con un par de coordenadas pseudo-polares (r, θ), donde $r \in [0, 1]$ representa la distancia del punto al contorno de la pupila y $\theta \in [0 \dots 2\pi)$ el ángulo respecto al centro del iris. Este proceso se puede ver en la figura 5.4.

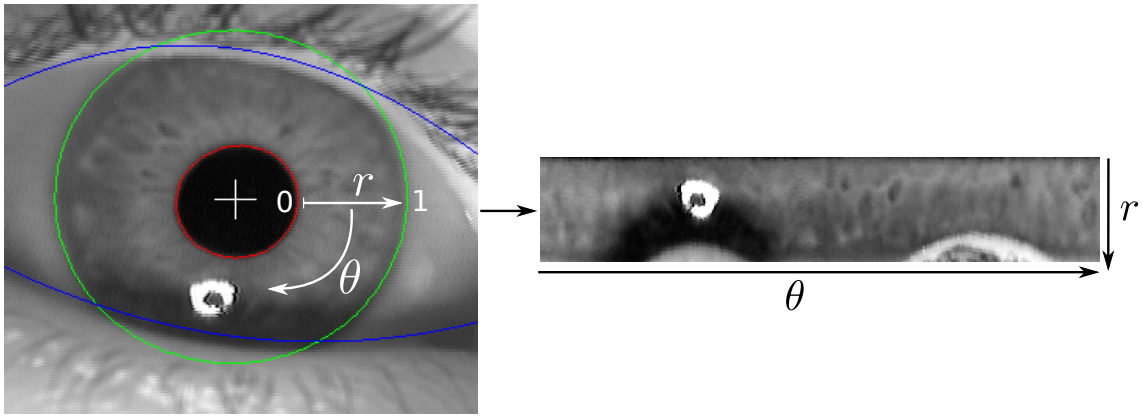


Figura 5.4: Esquema de la normalización del iris: luego de ser segmentado, la textura del iris se transforma en una imagen rectangular de dimensiones fijas. Se puede ver en la imagen rectangular las regiones afectadas por los párpados y el relfejo del iluminador infrarrojo.

Cada fila de la nueva imagen representa un "anillo" del iris y cada columna un ángulo en el anillo. Como la nueva imagen tiene dimensiones constantes, esto permitirá que todos los iris sean representados con un tamaño fijo, de manera que el sistema sea invariante frente a cambios en el diámetro de la pupila o del iris. Este modelo para representar el iris se conoce en la literatura como el modelo *rubber sheet*, o "lámina de goma", ya que el proceso equivale a extraer la textura del iris y deformarla para que quede rectangular.

Además, para cada punto en la imagen normalizada, se genera una máscara de ruido, que marca aquellos puntos que fueron afectados por los párpados y el iluminador infrarrojo[2] (figura 5.5).

[2] En las pruebas realizadas, el ruido generado por las pestañas no fue significativo por lo que las mismas no se consideran.

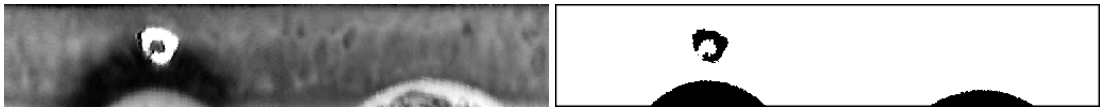


Figura 5.5: Textura normalizada y máscara de ruido. Las partes negras representan los puntos que fueron afectados por los párpados y el iluminador

5.2.2. Codificación

Como se dijo antes, se utilizan los filtros de Log-Gabor unidimensionales para codificar la textura normalizada del iris. Cada fila de la textura normalizada es filtrada con un conjunto de filtros de Log-Gabor diseñados para tener una buena cobertura del espacio de las frecuencias (figura 5.6).

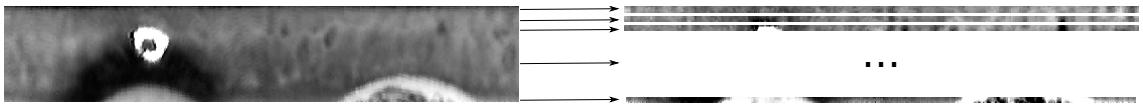


Figura 5.6: Extracción de señales unidimensionales de la textura del iris. Cada señal es un vector de valores que será filtrado y codificado por separado.

Para cada punto en la imagen, la respuesta de dicho punto al filtro es un punto complejo, donde la parte real se corresponde con la respuesta al filtro simétrico par y la parte compleja se corresponde con la respuesta al filtro simétrico impar (figura 5.7). Se codifica únicamente la fase de la respuesta, ya que, como se mencionó, la amplitud no aporta mucha información y es inestable frente a cambios de iluminación. El valor de la fase se cuantiza a uno de cuatro posibles valores, correspondientes al cuadrante de la fase, por lo que la respuesta de cada punto a cada filtro se codifica con dos bits. La figura 5.8 muestra este proceso.

La ventaja de este método es que si, por efecto del ruido, la fase se desplaza de cuadrante en dos imágenes del mismo iris, únicamente cambiará un bit del código, haciéndolo más resistente frente al ruido (es necesario un cambio de fase de casi 180^o para cambiar los dos bits del código correspondientes a ese punto).

El resultado de este procedimiento es una matriz binaria, donde cada fila de la matriz representa la codificación de una fila de la textura del iris. En el capítulo 7 se hará un análisis detallado de los parámetros utilizados en el filtro.

Invarianza frente rotaciones

La representación del código como una matriz binaria tiene una ventaja sumamente importante: permite, de manera sencilla, cierto nivel de invarianza frente a las rotaciones del ojo.

Una rotación del ojo se traduce en la textura del iris en un desplazamiento hacia un costado (figura 5.9). Por lo tanto, el código generado para el iris también estará desplazado hacia un costado respecto al código generado para el mismo ojo sin rotación. Esto será de suma utilidad en la etapa de matching, como se explicará en la sección 5.2.3.

Área del iris

Para minimizar la influencia de las pestañas y el párpado superior, se decidió excluir el cuarto superior del iris en la codificación. Además, ya que la textura del iris suele ser muy pobre en la región cercana al borde con la esclera, solamente se utiliza una región que abarca el 75 % del radio del iris. En la figura 5.10 se puede ver la región de interés para el algoritmo de codificación.

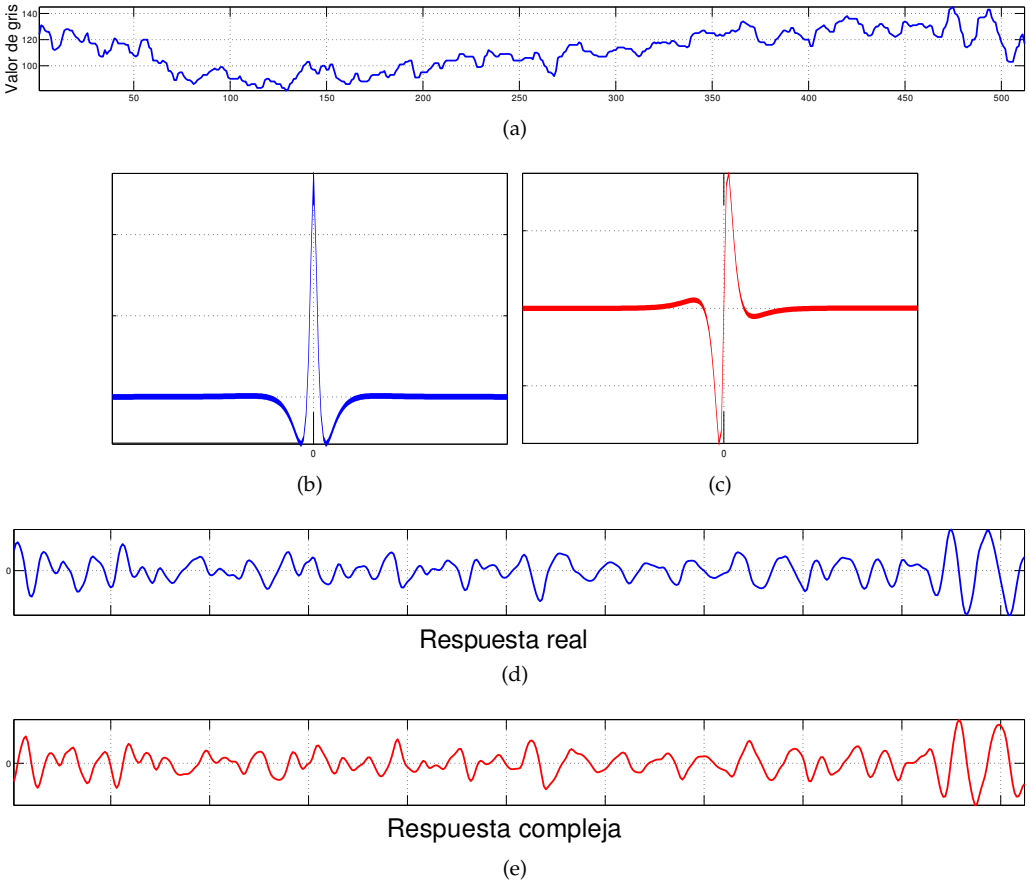


Figura 5.7: Proceso de codificación: 5.7a: señal de entrada (una línea de la textura del iris), 5.7b y 5.7c: filtro simétrico par e impar, respectivamente, 5.7d: respuesta de la señal al filtro simétrico par (corresponde a la parte real de la respuesta), 5.7e: respuesta de la señal al filtro simétrico impar (corresponde a la parte compleja de la respuesta)

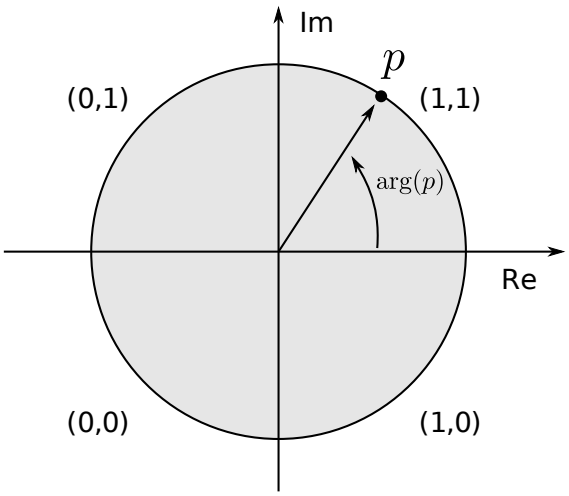


Figura 5.8: Cuantización de fase: el argumento (fase) de p cae en el primer cuadrante, por lo cual el valor de p se codifica como 11.

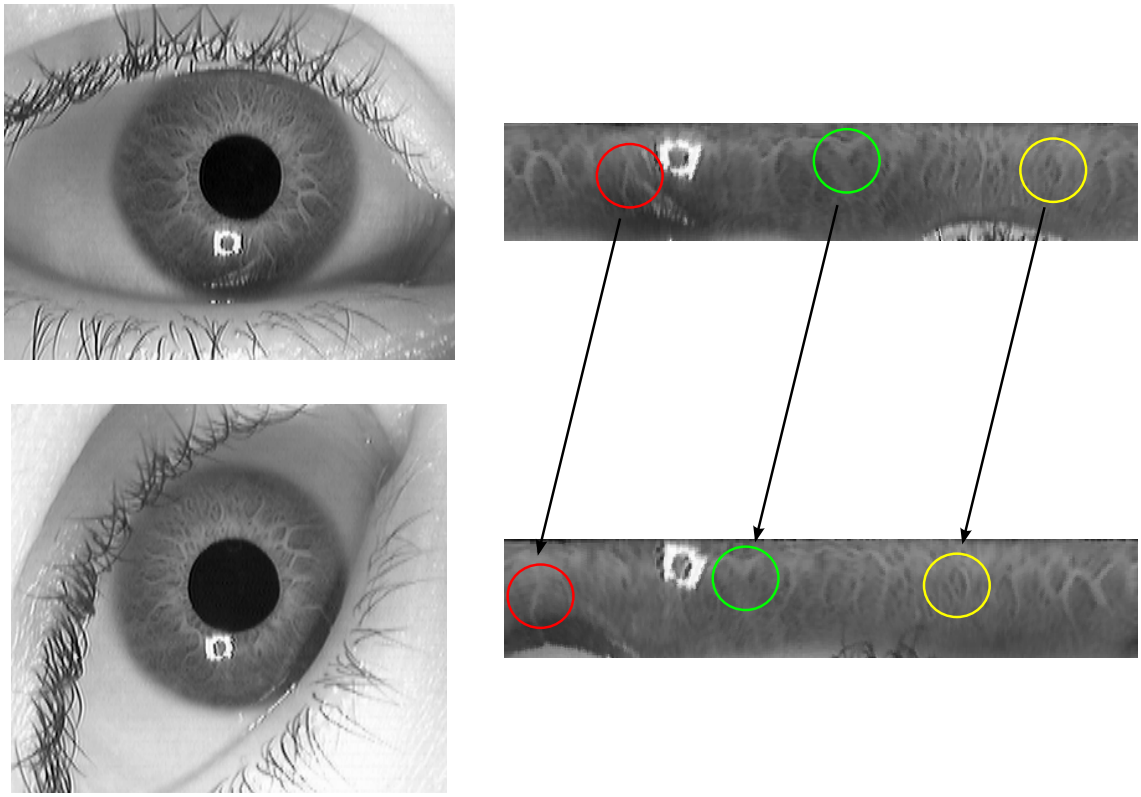
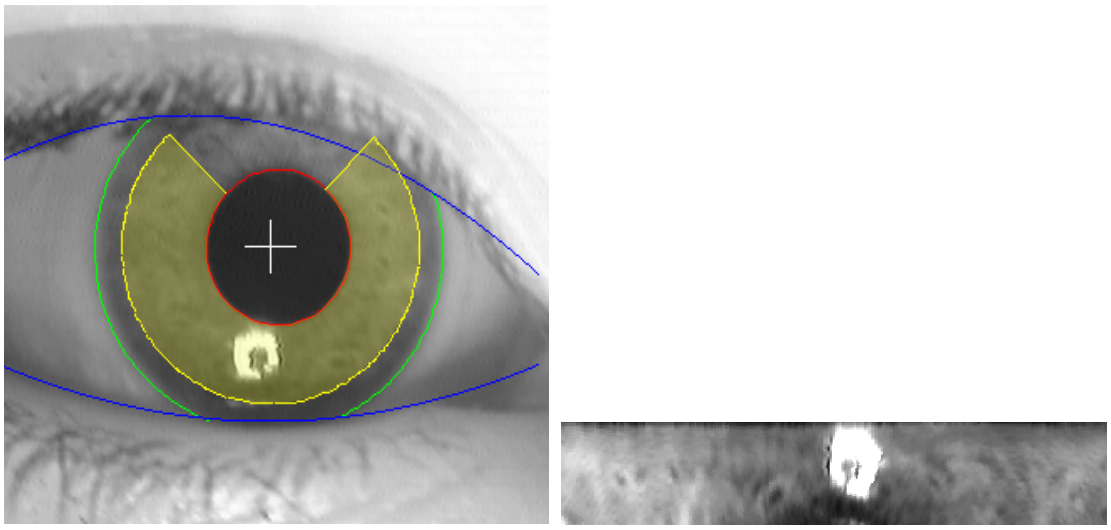


Figura 5.9: Efecto de la rotación del ojo: las características marcadas se desplazaron hacia un costado (en este caso, hacia la izquierda). Esto permitirá en la etapa de matching lograr invarianza frente a rotaciones simplemente haciendo un shift de un código hacia los costados.



(a) (b)

Figura 5.10: Región de interés para el algoritmo de codificación. La parte superior se excluye para evitar la obstrucción de las pestañas y los párpados, mientas que se utiliza un 75 % del radio del iris. En 5.10b se puede ver la región de interés normalizada

5.2.3. Matching

Como en la mayoría de los trabajos que utilizan códigos binarios, se decidió utilizar la distancia de Hamming para comparar dos códigos de iris. La ventaja principal de este método es que es sumamente rápido, ya que se pueden "empaquetar" varios bits del código en una palabra del procesador y realizar varias comparaciones en paralelo (por ejemplo, en un procesador común de 32 bits, se pueden realizar 32 comparaciones por ciclo de reloj).

Como métrica, la distancia de Hamming permitirá establecer qué tanto se *parecen* dos códigos de iris. Al haber tantas variables en el proceso previo a la codificación (pequeñas diferencias en la segmentación, variables de captura como iluminación, reflejos, distancia a la cámara, ruido electrónico en la cámara, etc.) es de esperar que dos códigos generados a partir del mismo iris no sean exactamente iguales, lo que daría como resultado una distancia de 0. El resultado de la comparación dará un valor que permitirá realizar una decisión acerca de si los dos códigos fueron generados a partir del mismo iris o no.

Invarianza frente a rotaciones

Como se mencionó antes, uno de los problemas a solucionar en la etapa del matching es la de la rotación del ojo. Ésta puede ocurrir en forma involuntaria por parte del usuario; una rotación de apenas un grado respecto a otra captura dará como resultado un código desfasado, y esto puede dar como resultado de la comparación una distancia de Hamming muy elevada.

Por ejemplo, suponiendo (en forma simplificada) que se tienen dos códigos de iris A y B de 10 bits cada uno generados por el mismo iris, con $A = 1011010001$ y $B = 0100100011$. Calculando la distancia de Hamming, se tiene:

$A =$	1	0	1	1	0	1	0	0	0	1
$B =$	0	1	0	0	1	0	0	0	1	1
$A \otimes B =$	1	1	1	1	1	1	0	0	1	0

con lo que la distancia de Hamming resultante es $7/10 = 0{,}7$, es decir, ambos códigos son desiguales en un 70 %. Esto es una distancia demasiado grande como para considerarse que los códigos fueron generados por el mismo iris. Sin embargo, si se define $B_{\rightarrow}$ como el código B pero rotado un bit a la derecha, se tiene lo siguiente:

$A =$	1	0	1	1	0	1	0	0	0	1
$B_{\rightarrow} =$	1	0	1	0	0	1	0	0	0	1
$A \otimes B_{\rightarrow} =$	0	0	0	1	0	0	0	0	0	0

Ambos códigos difieren en este caso en únicamente un bit, con lo que es razonable suponer que B fue generado por el mismo iris que A pero rotado levemente.

5.3. Resumen

En este capítulo se vio un resumen de las técnicas de codificación más comunes en la literatura. Se analizaron también algunas técnicas de matching entre códigos de iris. La cantidad de trabajos realizados en este aspecto es muy grande, aunque la mayoría suele derivar de alguna u otra forma las técnicas descriptas. Finalmente, se analizó en detalle la opción implementada. En el capítulo 7 se verá qué parámetros utilizar para obtener resultados óptimos.

Capítulo 6

Sistema implementado

En este capítulo se darán algunos detalles del sistema de reconocimiento de iris implementado para este trabajo. El sistema armado es un prototipo con todas las funcionalidades de un sistema completo, y consiste en dos subsistemas principales independientes: por un lado se encuentra el sistema de captura, y. por el otro, el software encargado de procesar las imágenes. La figura 6.1 presenta un esquema con los componentes del sistema, que serán descriptos a continuación.

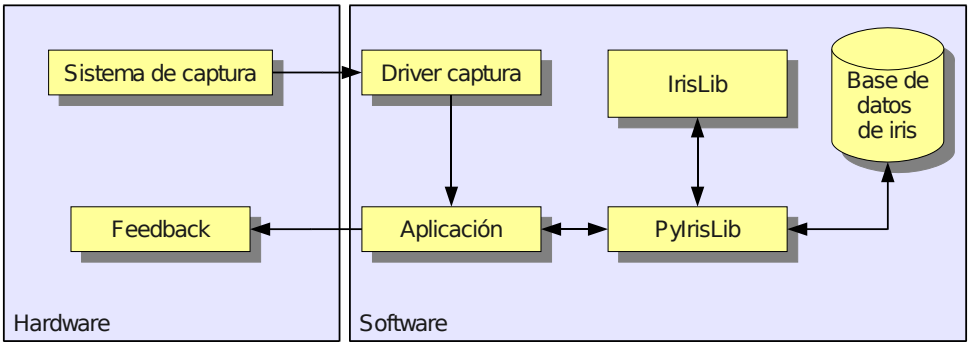


Figura 6.1: Componentes del sistema implementado

6.1. Hardware

El hardware del sistema está compuesto por el sistema de captura y una interfaz entre el sistema de captura y el software. Cuenta también con una unidad de *feedback* que le provee información al usuario en tiempo real.

6.1.1. Sistema de captura

Como se explicó en el capítulo 2, el sistema de captura consiste principalmente de una cámara y el lente. La cámara utilizada en el sistema implementado es una cámara del tipo CCTV (*closed-circuit TV*) con un sensor CMOS utilizada para vigilancia, que tiene capacidades de captura en el infrarrojo cercano. El sensor captura imágenes en blanco y negro, por lo que tiene una mayor sensibilidad a la luz que las cámaras a color.

El sistema cuenta con un arreglo circular de LEDs infrarrojos, que se ubica en la parte inferior de la cámara. Esto permite capturar imágenes del iris con mayor contraste, pero tiene la desventaja

que el reflejo del iluminador en el ojo se produce justo en el área del iris, y en algunos casos el párpado inferior puede producir sombra.

6.1.2. Interfaz con el software

El sistema de captura está conectado con el software de reconocimiento por medio de una placa capturadora (framegrabber) que digitaliza la señal análoga de la cámara. La cámara está conectada al framegrabber por medio de un cable RCA común.

El sistema fue probado con dos tipos de framegrabbers *commercial-off-the-shelves* (COTS, es decir, disponibles en cualquier negocio) de bajo costo, un framegrabber PCI y uno con conectividad USB. Ambos son capaces de producir imágenes de una resolución de 720×578 pixels.

6.1.3. Feedback

La unidad de feedback consiste en un monitor de computadora que le permite ver al usuario el video de la cámara en vivo. El sistema indica gráficamente algunos eventos; por ejemplo, cuando el usuario está bien posicionado y se captura una imagen del iris, se muestra en la pantalla un destello que le indica al usuario que se capturó una imagen exitosamente y aparece también en la pantalla la imagen capturada.

6.2. Software

El hardware se comunica con la aplicación principal, y es esta aplicación la que contiene la lógica que conforma el sistema de reconocimiento de iris.

6.2.1. Aplicación

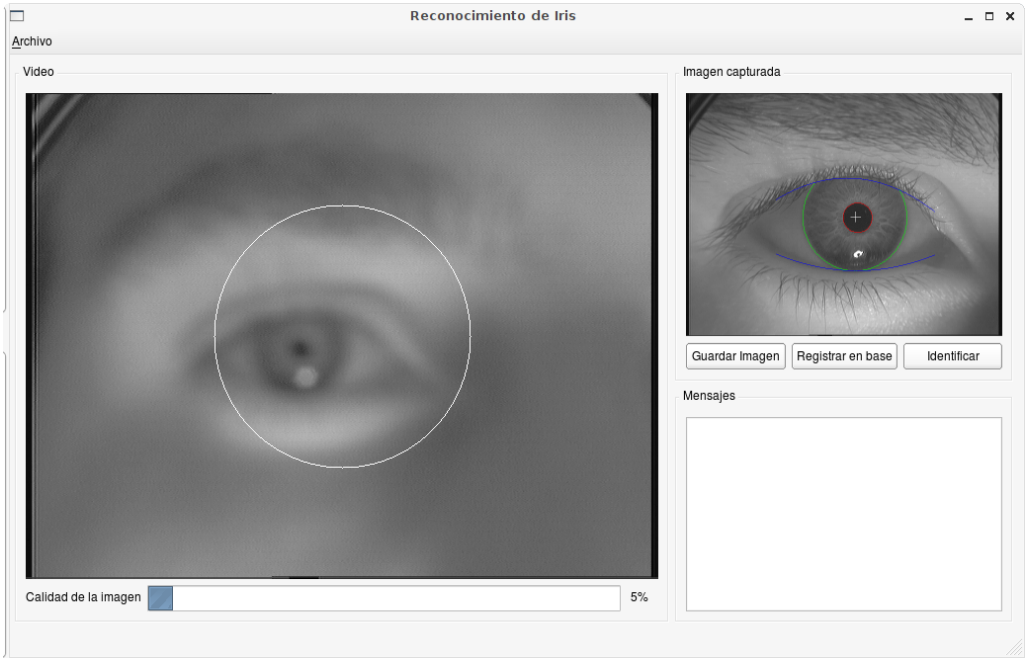
La aplicación principal consiste en una interfaz gráfica de usuario (GUI) que muestra el video en vivo y se encarga de interactuar con el sistema de captura. También interactúa con el sistema de reconocimiento propiamente dicho, que se encuentra encapsulado en una librería. En la figura 6.2 se puede ver una captura de pantalla de la aplicación en funcionamiento. Esto es lo que ve el usuario del sistema y forma parte del feedback presentado al usuario.

6.2.2. Librería IrisLib

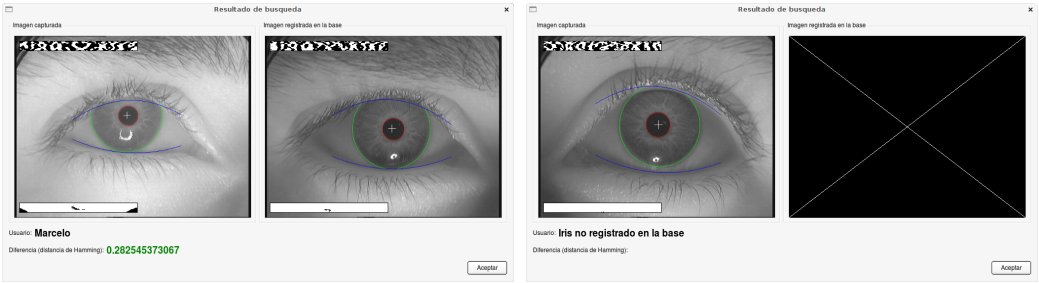
En el corazón del sistema, se encuentra la librería desarrollada que contiene todos los algoritmos descriptos en este trabajo, bautizada con el nombre de **IrisLib**. La librería posee tres módulos principales, cada uno independiente del otro y dedicado a una función en particular:

- Módulo de **procesamiento de video**: se encarga de recibir el stream de video del sistema de captura y de decidir cuándo se capturó una imagen apta para la identificación.
- Módulo de **segmentación**: se encarga de la segmentación de las imágenes.
- Módulo de **codificación**: genera los códigos de iris a partir de las imágenes previamente segmentadas.

La librería está implementada en el lenguaje C++, lo que permite optimizar los recursos del procesador para funcionar en un entorno en tiempo real. Para las operaciones relacionadas con procesamiento de imágenes utiliza la librería OpenCV, disponible libremente. La librería IrisLib posee una API que permite acceder a las funcionalidades descriptas.



(a)



(b) (c)

Figura 6.2: Interfaz gráfica de la aplicación: 6.2a: pantalla principal de la aplicación. La imagen de la izquierda es el video en tiempo real, y la imagen pequeña de la derecha es la última imagen capturada con la segmentación superpuesta. El círculo blanco en la imagen izquierda sirve para que el usuario se posicione correctamente. La barra inferior indica la calidad de la imagen. 6.2b: resultado de una búsqueda en la base de datos: en la izquierda, la imagen capturada y en la derecha la imagen correspondiente al iris almacenado en la base de datos. En ambas imágenes se superpone el código de iris en la parte superior y la máscara de ruido en la parte inferior. 6.2c: resultado de una búsqueda de un iris inexistente en la base de datos.

6.2.3. PyIrisLib

Para simplificar el desarrollo, IrisLib posee también una interfaz con el lenguaje Python bautizada **PyIrisLib**. Esto permite el desarrollo rápido de aplicaciones de prueba que utilizan dicha API con muy pocas líneas de código (en el apéndice A se muestran algunos ejemplos de cómo se utiliza esta librería). PyIrisLib provee además la funcionalidad extra de almacenamiento y búsqueda de códigos de iris en una base de datos externa (actualmente implementado utilizando el motor de bases de datos SQLite).

El apéndice A contiene información detallada sobre la estructura de la librería, la API y su utilización.

Capítulo 7

Resultados

En este capítulo se analizarán los resultados obtenidos a partir de los algoritmos descriptos. Los resultados se pueden dividir en dos categorías diferentes: los resultados de los algoritmos de segmentación y los resultados de la codificación.

7.1. Resultados de la segmentación

Para medir el resultado del algoritmo de segmentación y compararlo con otros se tuvieron en cuenta tres factores: el tiempo de procesamiento, la flexibilidad del algoritmo y el porcentaje de imágenes correctamente segmentadas.

Para probar los algoritmos, se utilizaron dos bases de datos de iris libremente disponibles: la base de datos CASIA-1 y la base de datos MMU [LE04]. Estas bases fueron elegidas ya que las imágenes contenidas en las mismas son similares a las imágenes obtenidas por el sistema de captura desarrollado. En otras bases de datos (por ejemplo, CASIA-3), el reflejo del iluminador infrarrojo es muy intenso y se encuentra dentro de la pupila, lo que puede producir problemas en el algoritmo de ajuste de contorno de la pupila, explicado en la sección 3.2.2[1].

La base de datos CASIA-1 tiene un total de 756 imágenes, mientras que la MMU tiene 450. En total, se probó la segmentación sobre 1206 imágenes. Las pruebas se hicieron en una computadora con un procesador Intel de 1.4 GHz. Los resultados, comparados con los resultados de otros trabajos, se pueden ver en el cuadro 7.1.

Trabajo	Cant. de imágenes	Resultados	Tiempo promedio
Masek [Mas03]	756 (CASIA-1)	83 %	$>$ 3 seg.
Cui [CWT$^+$] (*)	756 (CASIA-1)	99,54 %	0,24 seg.
Yuan [YZL06] (*)	756 (CASIA-1)	99,6 %	0,146 seg.
Lili [LM05]	2400	99,75 %	N/D
Propuesto (*)	1206	98,17 %	0,03 seg.
Propuesto (con párpados)	1206	94,8 %	0,04 seg.

Cuadro 7.1: Comparación de los resultados de la segmentación. (*) no segmenta párpados

Si bien el algoritmo implementado no tiene los mejores resultados en cuanto al total de las

[1] Hasta el momento no se compiló con el sistema de captura diseñado una base de datos lo suficientemente grande como para hacer pruebas significativas, por lo que se decidió utilizar una base de datos preexistente con imágenes con características similares a las obtenidas por el sistema de captura.

imágenes segmentadas correctamente, los resultados obtenidos fueron muy buenos (aún considerando los párpados) y, además, el algoritmo resultó ser el más rápido de todos.

El algoritmo de segmentación también provee resultados muy buenos con el sistema de captura. En el sistema implementado, el algoritmo de segmentación está ligado con el proceso de selección de buenas imágenes, ya que se segmenta cada frame de video en tiempo real y a partir del resultad de las segmentación se decide sobre la calidad de la imagen, como se vio en el capítulo 4.

En la figura 7.1 se pueden ver los resultados de la segmentación de varias imágenes.[2]

7.2. Resultados de la codificación

La codificación junto con el matching son los métodos que, a fin de cuentas, permitirán determinar si dos iris son iguales o no. Esto a su vez es lo que permite decidir acerca de la identidad de una persona: si dos iris son considerados *iguales*, se asumirá que provinieron del mismo ojo, y, por lo tanto, de la misma persona.

Entonces, un algoritmo de codificación debe ser capaz de generar un código que permita diferenciar un iris de otro. Antes de hacer esto, es necesario preguntarse cuál es el *poder de discriminación* del iris humano, es decir, qué tanta información posee que permita diferenciar un iris de otro.

Este problema fue analizado en profundidad por Adler [AYL06] y Dembinsky [Dem06], llegando a la importante conclusión que la cantidad de información de una característica biométrica, si bien acotada, en general está *ligada a la elección de features*. En el caso del iris, esto quiere decir que la cantidad de información discriminativa del iris está dada exclusivamente por el método de codificación utilizado, y que no existe una medida universal que permita definir la cantidad de información del iris.

7.2.1. Verificación vs. Identificación

Para realizar las pruebas, es necesario definir primero dos procesos distintos que se llevan a cabo en un sistema de reconocimiento de iris: la *verificación* y la *identificación*.

La verificación consiste en comparar dos códigos de iris uno contra otro (comparación $1:1$) y, en base a su distancia, realizar una decisión acerca de la identidad: o bien ambos iris son iguales (si la distancia entre ambos códigos es menor a un umbral t), o son distintos (si la distancia es mayor a t). Esto da lugar a cuatro resultados posibles:

- **Verdadera aceptación**: ocurre cuando dos iris son correctamente reconocidos como el mismo iris.
- **Verdadero rechazo**: ocurre cuando dos iris son correctamente reconocidos como dos iris distintos.
- **Falsa aceptación**: ocurre cuando dos iris son reconocidos como el mismo iris pero en realidad son dos iris distintos.
- **Falso rechazo**: ocurre cuando dos iris son reconocidos como iris distintos pero en realidad son el mismo.

La identificación, en cambio, consiste en comparar un código de iris contra una base de datos entera (comparación $1:n$) y decidir acerca de la *identidad* de dicho iris, es decir, a qué usuario registrado en la base pertenece. La identificación no es más que un *conjunto de verificaciones*: se hace una verificación entre el código de iris a identificar y cada iris en la base. De todas estas

[2] Muchas de las imágenes fueron extraídas de otras bases de datos a modo de prueba

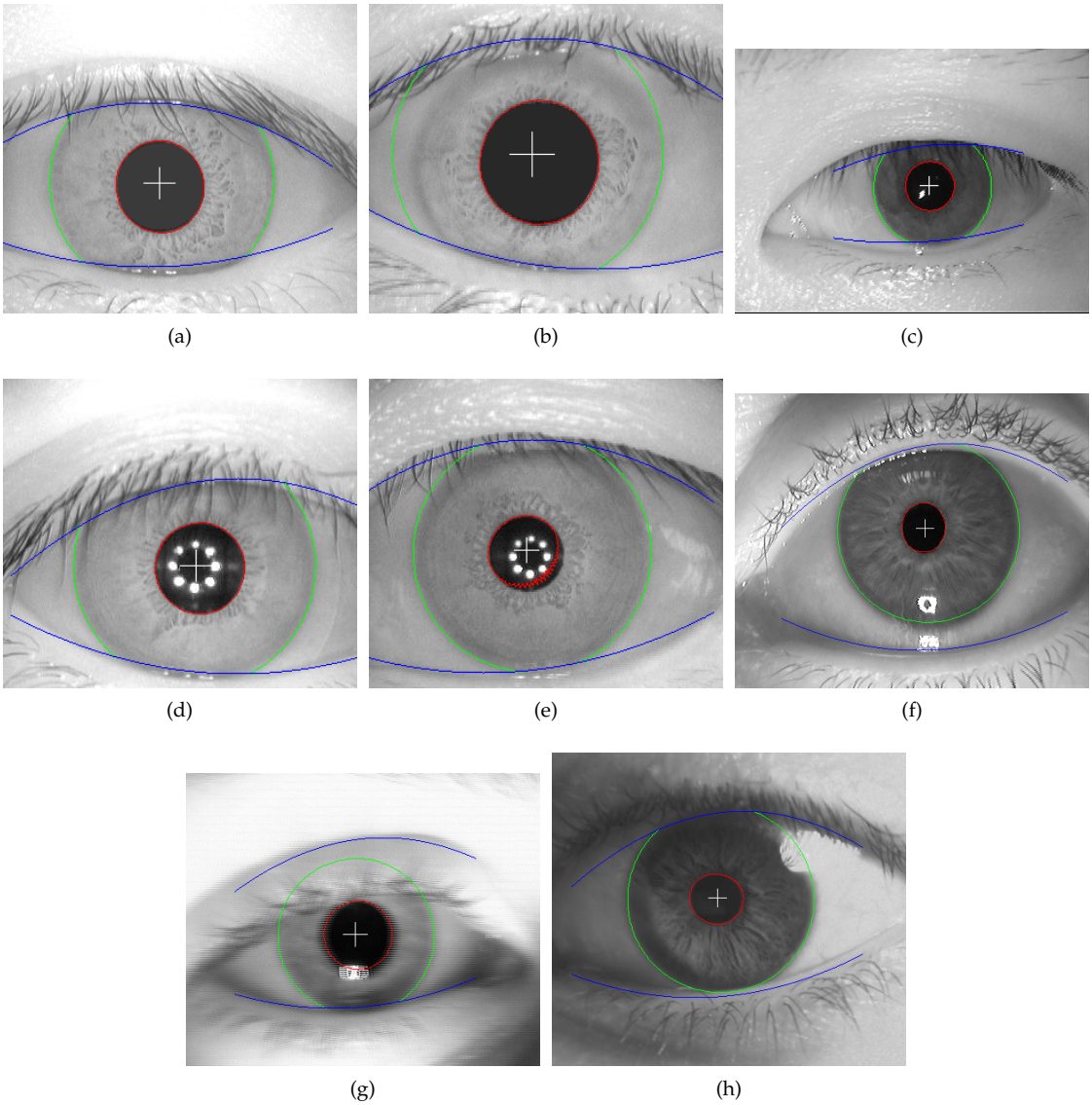


Figura 7.1: Resultado del algoritmo de segmentación aplicado a varias imágenes: 7.1a: Imagen de la base de datos CASIA-1. 7.1b: Imagen donde el algoritmo de segmentación falla en localizar correctamente el iris, esto se debe al bajo contraste entre el iris y la esclera. 7.1c: Imagen de la base de datos MMU. 7.1d: Imagen de la base de datos CASIA-3. 7.1e: Imagen de la base de datos CASIA-3 donde el reflejo del iluminador da resultados erróneos en la segmentación de la pupila (aunque el iris y los párpados son correctamente segmentados). 7.1f: Imagen obtenida por el sistema de captura implementado. 7.1g: Frame de video segmentado. La segmentación en esta etapa permite establecer la calidad de la imagen a partir de la calidad del borde entre la pupila y el iris, como se explicó en el capítulo 4. 7.1h: Imagen obtenida por el sistema de captura sin iluminación infrarroja. Aunque la calidad de la imagen no es buena y el iris carece de contraste, el algoritmo de segmentación dio buenos resultados. Esto muestra la flexibilidad del algoritmo en distintas situaciones de captura.

comparaciones, se toma una decisión en base al código más parecido registrado en la base: si la distancia entre ambos códigos es menor al umbral t, se identifica el iris como el correspondiente al código más parecido. Si, en cambio, la distancia es mayor a t, se decide que el iris a identificar

no se encuentra en la base de datos y se produce un rechazo.

Este proceso también da lugar a cuatro resultados posibles:

- **Identificación correcta**: ocurre cuando el iris es correctamente identificado en la base de datos.
- **Rechazo correcto**: ocurre cuando el iris a buscar no está registrado en la base de datos, y el sistema correctamente falla en encontrar dicho iris en la base.
- **Identificación errónea**: ocurre cuando un iris (que puede estar registrado en la base o no) es identificado erróneamente como otro iris registrado en la base.
- **Falso rechazo**: ocurre cuando la búsqueda falla en encontrar un código de iris previamente registrado en la base.

Evaluando la verificación

Las pruebas de codificación y matching se realizan sobre dos bases de datos de iris preexistente, la base CASIA-1 y la base MMU. Cada base contiene un conjunto de imágenes de ojos divididas en varias clases, donde cada clase se corresponde con un ojo en particular. Entonces, las comparaciones *intra-clase* son las comparaciones entre imágenes del mismo ojo mientras que las comparaciones *inter-clase* son comparaciones entre ojos distintos.

Para cada imagen, se genera el código de iris y se hace un matching contra todas las otras imágenes de la base. Entonces, para cada imagen, habrá un conjunto de comparaciones intra-clase y otro conjunto de comparaciones inter-clase. Repitiendo el proceso para todas las imágenes de la base, se tiene un conjunto de comparaciones inter-clase C_{inter} y otro de comparaciones intra-clase C_{intra}, donde el resultado de cada comparación es la distancia de Hamming entre los dos códigos de iris generados a partir de cada imagen.

A partir de estos dos conjuntos de distancias y de un umbral t, se definen los siguientes valores:

- La **Tasa de Falsa Aceptación**, o **FAR** a partir del inglés *false accept rate*: es el porcentaje de comparaciones que dieron como resultado una falsa aceptación, y se define como:

$$\text{FAR}(t) = 100\frac{|\{x \in C_{\text{inter}} : x < t\}|}{|C_{\text{inter}}|} \tag{7.1}$$

- La **Tasa de Falso Rechazo**, o **FRR** del inglés *false reject rate*: es el porcentaje de comparaciones que dieron como resultado un falso rechazo, y se define como:

$$\text{FRR}(t) = 100\frac{|\{x \in C_{\text{intra}} : x \geq t\}|}{|C_{\text{intra}}|} \tag{7.2}$$

Si se genera un histograma normalizado de los valores de ambos conjuntos, la FAR representa el área de la curva del histograma de C_{inter} a la izquierda de t mientras que la FRR es el área de la curva del histograma de C_{intra} a la derecha de t (figura 7.2).

De la FAR y la FRR se puede calcular un nuevo valor, la **tasa de error equivalente**, o **EER** del inglés *equal error rate*, y representa el valor en el cual la FAR se iguala a la FRR. El EER sirve como medida general que establece el error del sistema (mientras menor sea el EER, mejor performance tendrá el sistema). Un EER=0 representa un desempeño perfecto, es decir, que todos los pares de imágenes fueron clasificados correctamente.

También se puede medir la *separabilidad* o *decidiblidad* entre las dos clases [Dau04]: dadas la media de las distancias de las comparaciones intra-clase μ_{intra} y la media inter-clase μ_{inter} y sus

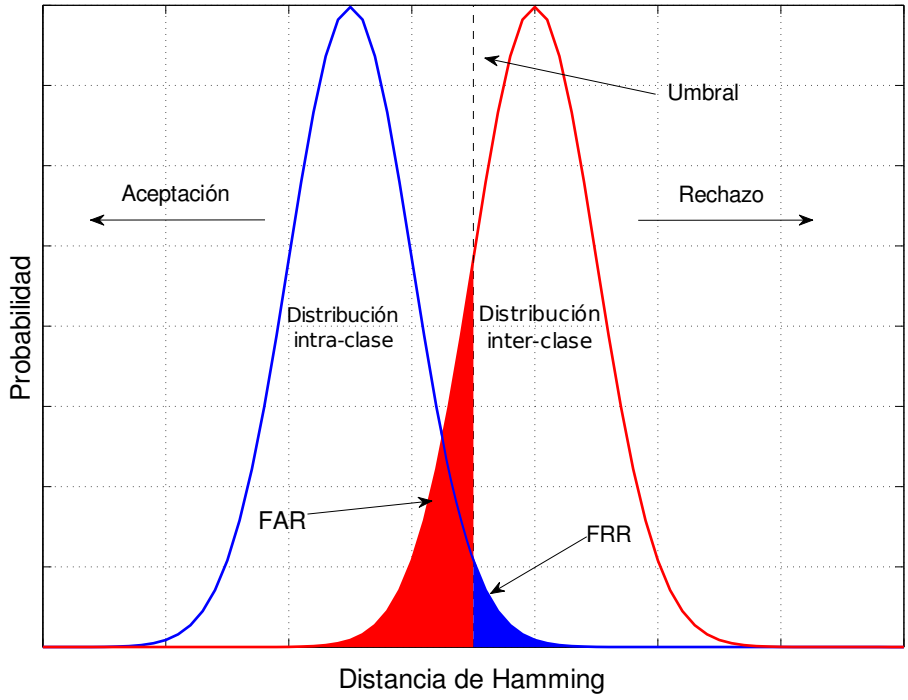


Figura 7.2: Ejemplo de distribución de comparaciones intra-clase e inter-clase. La FRR y la FAR representan el área bajo cada curva respecto al valor de threshold.

respectivas desviaciones estándar, σ_{intra} y σ_{inter}, se define la decidibilidad d' como:

$$d' = \frac{|\mu_{\text{intra}} - \mu_{\text{inter}}|}{\sqrt{\frac{\sigma^2_{\text{intra}} + \sigma^2_{\text{inter}}}{2}}} \tag{7.3}$$

Este es un valor escalar (sin dimensión) que mide la separación entre las dos clases, medido en desviaciones estándar. Mientras mayor sea la separación, mayor será el poder de discriminación del método de codificación.

Evaluando la identificación

La identificación se puede evaluar en forma paralela a la verificación. Para cada código de iris, se busca en el resto de la base de datos aquel que tenga la menor distancia de Hamming. Si ambos pertenecen a la misma clase, entonces se obtiene una identificación correcta, de lo contrario se tiene una identificación falsa. Si la menor distancia de Hamming es superior al umbral t, se produce un rechazo, que es falso si en la base de datos existe otro código perteneciente al mismo iris.

Sumando los falsos rechazos y las falsas identificaciones, se tiene como resultado el error total en la identificación.

7.2.2. Parámetros del método de codificación y matching

Como se vio en el capítulo 5, se utilizan los filtros de Log-Gabor unidimensionales para codificar la textura del iris, definidos en el espacio de las frecuencias por:

$$\hat{G}(f) = \exp\left(\frac{-\log(f/f_0)^2}{2\log(\sigma/f_0)^2}\right) \tag{7.4}$$

Los parámetros utilizados en el método de codificación son entonces:

- La resolución angular de la textura del iris
- La resolución radial de la textura del iris
- La frecuencia central f_0 de cada filtro
- El ancho de banda de cada filtro, definido por el cociente σ / f_0

Para el matching, el único parámetro es la cantidad de rotaciones del código para compensar por la rotación del ojo. La cantidad de rotaciones debe permitir un balance entre flexibilidad y velocidad, ya que muchas rotaciones permiten un mayor ángulo de rotación del ojo pero conlleva demasiadas comparaciones. En todas las pruebas se hicieron rotaciones de 20 pixels a la derecha y 20 a la izquierda, con un salto de 2 pixels. Por lo tanto, entre dos códigos, se hacen un total de 21 comparaciones (10 rotando a la izquierda, 10 a la derecha y una sin rotar).

7.2.3. Resultados para la base de datos CASIA-1

Esta base de datos posee un total de 756 imágenes, con un total de 108 ojos distintos y 7 imágenes por ojo. Luego de segmentar todas las imágenes, se descartaron 41 imágenes por errores de segmentación y por baja calidad, por lo que se utilizaron un total del 715 imágenes. Se hicieron entonces un total de 2.077 comparaciones intra-clase y 253.178 comparaciones inter-clase por cada prueba, y se hicieron varias pruebas para encontar los parámetros óptimos del filtro.

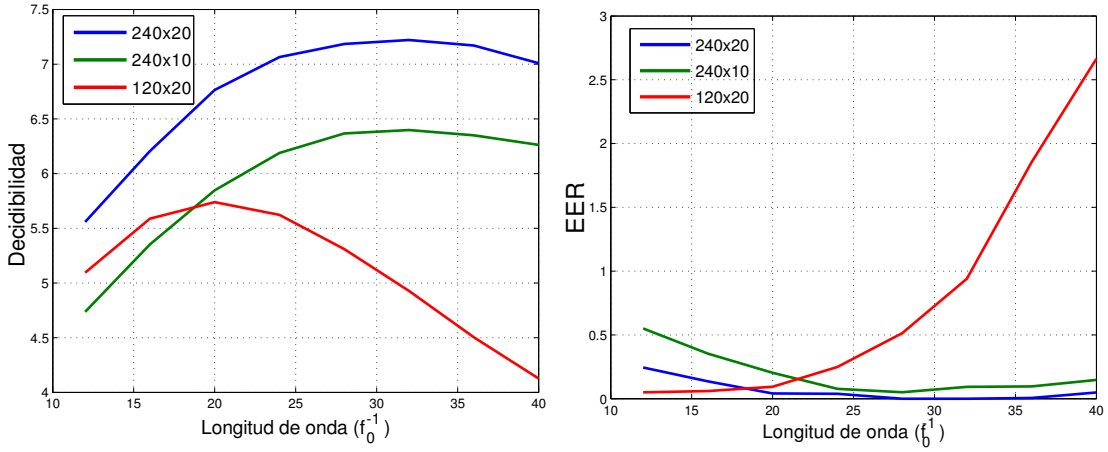
En la primer prueba, se utiliza un ancho de banda de aproximadamente dos octavas, dadas por el coeficiente $\sigma / f_0 = 0{,}5$. Se probaron distintos valores para f_0 y tres distintos tamaños de código, definidos como *resolución angular* $\times$ *resolución radial*. A diferencia de Masek [Mas03], se utilizó únicamente el resultado del filtro simétrico par del filtro de Log-Gabor (es decir, la parte real del filtro) con el fin de reducir el tamaño del código, en principio, a la mitad. En el cuadro 7.2 se pueden ver los resultados de estas pruebas.

Tamaño / f_0^{-1}	240×20		240×10		120×20	
	EER (%)	Decidib.	EER (%)	Decidib.	EER (%)	Decidib.
12	0.24	5.56	0.55	4.74	0.05	5.09
16	0.14	6.21	0.35	5.35	0.06	5.59
20	0.04	6.76	0.2	5.85	0.09	5.74
24	0.04	7.07	0.08	6.19	0.25	5.62
28	0	7.18	0.05	6.37	0.51	5.31
32	**0**	**7.22**	0.09	6.4	0.94	4.93
36	0.01	7.17	0.1	6.35	1.85	4.5
40	0.05	7.01	0.15	6.26	2.67	4.13

Cuadro 7.2: EER y decidiblidad para distintos parámetros del código, utilizando la parte real del filtro de Log-Gabor con un valor de $\sigma / f_0 = 0{,}5$

Se pueden ver muy buenos resultados sobre las imágenes analizadas, obteniendo resultados perfectos para una resolución de 240×20 y frecuencia central de $1/28$ y $1/32$ pixels (es decir, en todas las comparaciones se decidió correctamente si dos iris eran iguales o no). Un umbral de 0,3775 para la distancia de Hamming permite dividir las dos clases. Esto quiere decir que no hay dos comparaciones intra-clase que de una distancia de Hamming superior a 0,3775, ni dos comparaciones inter-clase con una distancia menor a 0,3775. La figura 7.4 muestra el histograma de las comparaciones intra-clase e inter-clase para código generado con estos parámetros.

A fin de ver si 0,5 es el mejor valor para el coeficiente σ / f_0, se realizó una prueba similar fijando los parámetros del código en un tamaño de 240×20 y un valor para f_0 de $1/32$. La figura



(a) (b)

Figura 7.3: 7.3a Decidibilidad para distintos valores de f_0 y distintos tamaños de código. 7.3b Ídem EER.

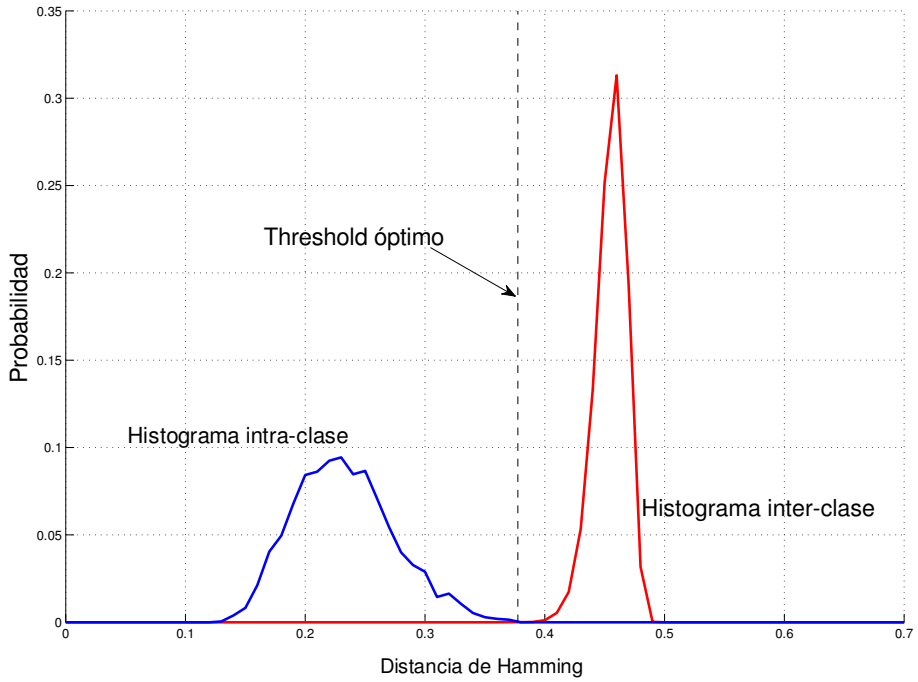


Figura 7.4: Histograma de distancias de Hamming de comparaciones intra-clase e inter-clase para un tamaño de código de 240×20, con $f_0 = 1/32$ y $\sigma/f_0 = 0{,}5$. Un umbral de 0,3775 permite una división perfecta de las dos clases, donde el error es nulo.

7.5 muestra los resultados de esta prueba, y se puede ver que la decidibilidad es máxima justo en este valor (aunque no por mucho). Se puede decir entonces que con una frecuencia central de $1/32$ y un ancho de banda de aproximadamente dos octavas, este filtro tiene resultados óptimos en esta base de datos.

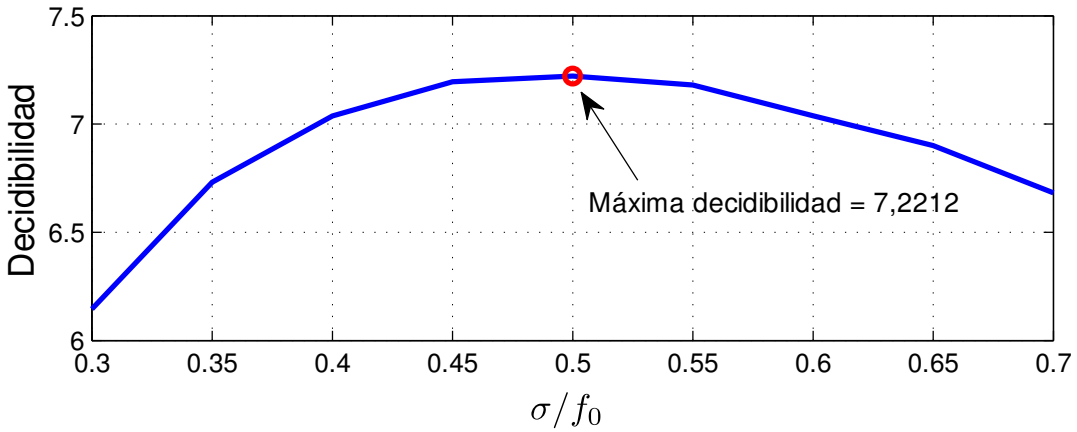


Figura 7.5: Decidibilidad para distintos valores del parámetro σ/f_0, con un código de tamaño 240×20 y $f_0 = 1/32$. El valor óptimo se encuentra en $\sigma/f_0 = 0{,}5$.

Resultados de la identificación

Una vez definidos los parámetros del código, se realiza la identificación de todas las imágenes de la base. Como se obtuvo un error del 0 % en la verificación, el error en la identificación también es del 0 % ya que todas las comparaciones dan el resultado correcto y es imposible obtener un falso rechazo o una falsa aceptación para las imágenes analizadas.

Si bien el error en la identificación fue del 0 % en las pruebas realizadas, esto no quiere decir que sea imposible realizar una falsa identificación. En la sección 7.3 se verá cómo se puede estimar la probabilidad de cometer un error en la identificación.

7.2.4. Resultados para la base de datos MMU

Esta base de datos tiene un total de 450 imágenes, y la cantidad de imágenes por ojo es variable. De las 450 imágenes, se descartaron un total de 8 por errores de segmentación y mala calidad. Las imágenes de la base son de mucho menor calidad y resolución que en la base de datos CASIA-1 por lo que es razonable suponer que los resultados con los mismos parámetros de codificación quizás no serán óptimos. Se hicieron un total de 869 comparaciones intra-clase y 96.592 comparaciones inter-clase.

Para empezar, a diferencia del caso anterior, se fijó, partiendo de los resultados obtenidos por [Mas03], un tamaño de 240×20 para el código y una frecuencia central f_0 de $1/32$. Se buscó entonces el parámetro σ/f_0 que maximice la decidibilidad. Como se puede ver en la figura 7.6, los resultados fueron bastante distintos a los obtenidos para la base CASIA-1. La mejor separabilidad se obtuvo con $\sigma/f_0 = 0{,}25$ y la separabilidad máxima fue de apenas 4,12 (a diferencia de antes donde se obtuvo una separabilidad de 7,22).

Una vez definido el valor del cociente σ/f_0, se buscó el valor óptimo de f_0 para un tamaño de código fijo de 240×20. La decidibilidad y el error para distintos valores de f_0 se pueden ver en el cuadro 7.3, y se puede ver que, nuevamente, la frecuencia $f_0 = 1/32$ maximizó la decidibilidad.

Aún así, estos valores de decidibilidad están lejos de los obtenidos para la base CASIA-1. Esta diferencia es causada por la baja resolución de las imágenes de esta base de datos, lo que demuestra la importancia de un buen sistema de captura.

La curva ROC

A diferencia de la base de datos CASIA-1, con la base MMU no se obtuvo una separación perfecta de las dos clases. Esto quiere decir que la elección de un umbral afectará el resultado: cada

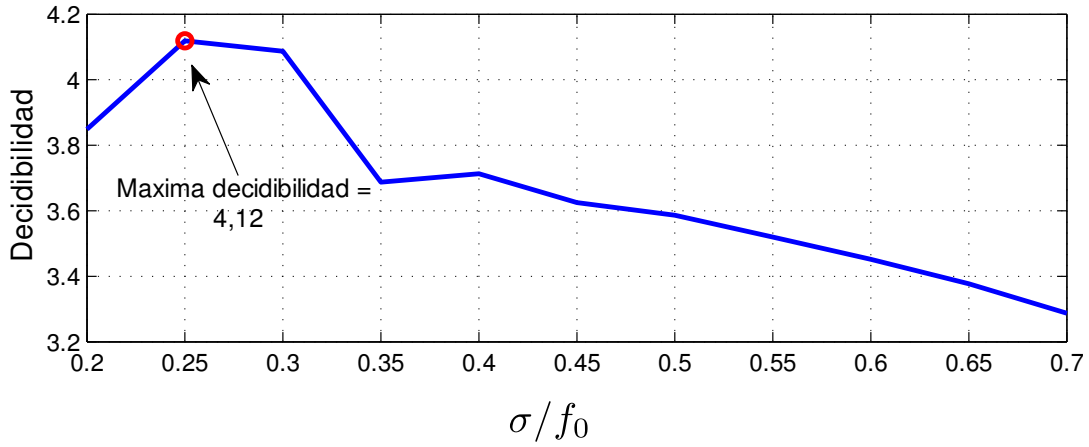


Figura 7.6: Decidibilidad en la base de datos MMU para distintos valores del parámetro σ / f_0, con un código de tamaño 240×20 y $f_0 = 1/32$. El valor óptimo se encuentra en $\sigma / f_0 = 0{,}25$.

f_0	EER	Decidibilidad
16	3,7803	3,8761
20	3,5733	4,0252
24	3,4433	4,0836
28	3,33	4,1084
32	**3,2637**	**4,1189**
36	3,4066	4,1177
40	3,5779	4,1122

Cuadro 7.3: Valores de error y decidibilidad para la base de datos MMU con un código de 240×20 y $\sigma / f_0 = 0{,}25$

umbral t genera un FAR(t) y un FRR(t), como se definió anteriormente. Graficando los puntos (FAR(t), FRR(t)) para distintos valores de t se obtiene una curva denomianda curva ROC, del inglés *receiver operating characteristic*. Esta curva permite establecer en forma visual cuál será la performance del sistema bajo distintos valores de t. En la figura 7.7 se puede ver la curva ROC para la base MMU con los parámetros óptimos del código. El punto de intersección de la curva con la recta $y = x$ se corresponde con el EER del sistema.

Eligiendo valores de t tal que uno se posiciona arriba de la recta $y = x$ (es decir, aquellos valores que resultan en FAR $<$ FRR, correspondientes a un umbral bajo), se estará definiendo un entorno *conservador*, donde una falsa aceptación es considerada más grave que un falso rechazo. En forma inversa, debajo de la recta $y = x$ (es decir, FAR $>$ FRR) se está en un entorno *liberal*, donde se toleran más las falsas aceptaciones que los falsos rechazos.

Resultados de la identificación

De las 442 imágenes utilizadas, 12 de ellas (2,71 %) fueron rechazadas erróneamente, mientras que sólo una (0,23 %) fue identificada en forma errónea. En total, los errores en la identificación suman 2,94 %, un valor bastante bajo teniendo en cuenta la baja resolución de las imágenes presentes en la base de datos.

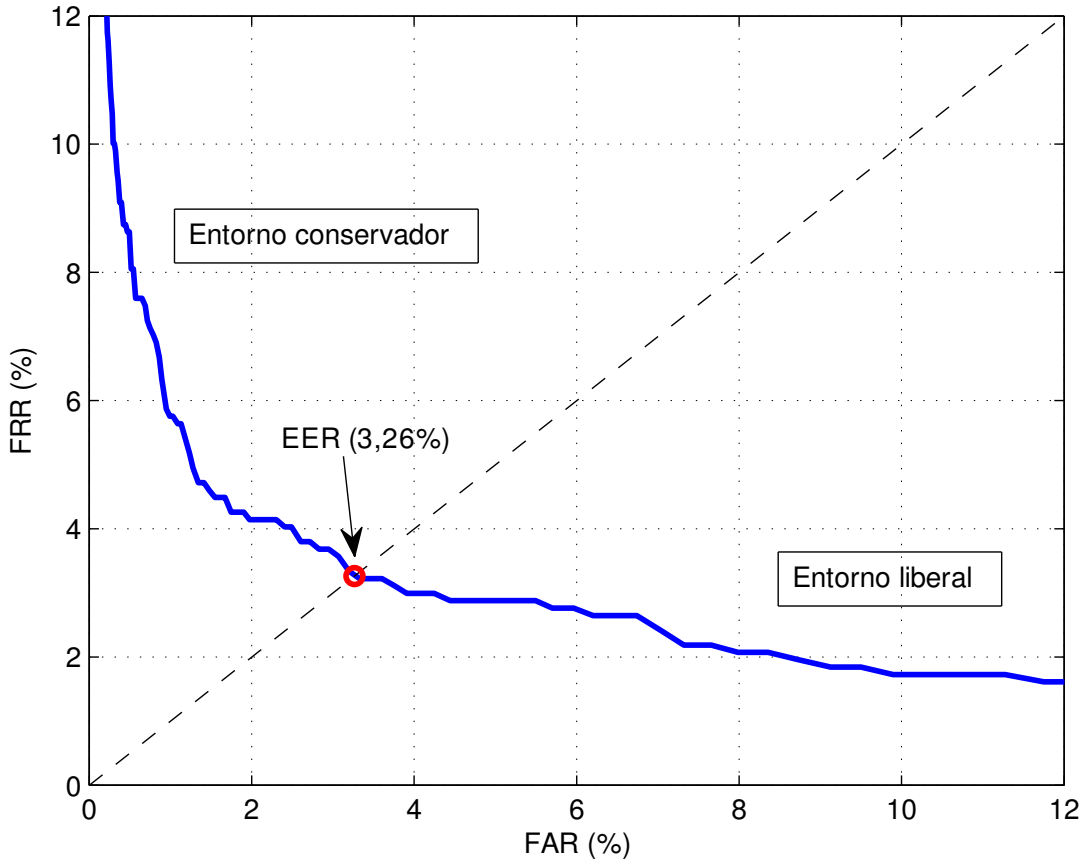


Figura 7.7: Curva ROC para la base de datos MMU

7.2.5. Comparación con otros trabajos

Entre los trabajos existentes, es posible hacer una comparación directa de los resultados de la codificación con los trabajos de Masek [Mas03] y Terissi [TCB06], ya que en dichos trabajos también se utiliza un filtro de Log-Gabor para hacer las pruebas y se usa la decidibilidad como herramienta para medir la performance del codificador (cuadro 7.4).

La mejora respecto a estos trabajos se obtuvo por dos motivos:

1. Un algoritmo de segmentación mejorado, que utiliza curvas en vez de círculos para modelar el contorno de la pupila y el iris.

2. La utilización de una parte del iris en vez del iris completo. Al utilizar el iris completo, se produce interferencia por parte de las pestañas y esto puede influir negativamente en el resultado final.

Trabajo	Base de datos	Decidibilidad
Masek	CASIA-1	6,1987
Terissi	Propia	5,0395
Propuesto	CASIA-1	**7,2212**

Cuadro 7.4: Comparación con trabajos similares

Una comparación con otros trabajos se puede ver en el cuadro 7.5. Se verifica que los resultados obtenidos en este trabajo son bastante alentadores. Sin embargo, las bases de datos utilizadas

en este trabajo son, en algunos casos, mucho menores a las utilizadas en los otros trabajos, por lo que queda pendiente analizar los resultados con bases de datos más grandes generadas con el sistema de captura desarrollado.

Trabajo	Identificación correcta (%)	EER (%)
Daugman [Dau01]	100	0,08
Huang [HMWT]	100	0,06
Ma [MTWZ04]	100	0,07
Monro [MRZ07]	100	N/D
Sanchez-Avila [SASR05]	99,6	0,12
Propuesto (CASIA-1)	100	0
Propuesto (MMU)	97,06	3,2637

Cuadro 7.5: Comparación con otros trabajos

7.3. Estimando la probabilidad de error en la identificación

Los resultados obtenidos para la base de datos CASIA-1 son óptimos en el sentido que se logró identificar cada iris en forma correcta en todos los casos. Sin embargo, esto no quiere decir que sea *imposible* realizar una identificación errónea. En esta sección se hará un análisis que permita definir, a partir de los resultados obtenidos, cuál es la probabilidad de realizar una identificación errónea.

La identificación suele ser la función más importante de un sistema biométrico. Mientras que en la verificación el resultado es únicamente una aceptación o rechazo de acuerdo a la distancia de Hamming, en la identificación se realizan varias comparaciones, cada una con su distancia de Hamming y con su probabilidad de error.

La probabilidad del falso rechazo en la *identificación* simplemente es igual a la probabilidad de falso rechazo en la *verificación*, ya que el resultado depende exclusivamente de la comparación entre los dos códigos del iris (el registrado en la base y el presentado al sistema).

En cambio, la probabilidad de una identificación errónea depende del tamaño de la base de datos: Si P_1 es la probabilidad de una falsa aceptación en la *verificación*, la probabilidad de obtener una falsa identificación en una base de datos de tamaño N, P_N es ([Dau00]):

$$P_N = 1 - (1 - P_1)^N \tag{7.5}$$

Es decir, la probabilidad de una falsa identificación crece en forma exponencial con el tamaño de la base se datos. Por ejemplo, para una probabilidad de falsa aceptación de apenas 0,01 %, la probabilidad de una identificación errónea en una base de datos de 1000 iris es de 9,5 %, lo cual es bastante elevado. Para una base de 30.000 personas, la probabilidad aumenta al 95 %. Esto hace que sea necesario una probabilidad de error extremadamente pequeña para una base de datos a gran escala.

7.3.1. Estimando la probabilidad real de una falsa aceptación

Para estimar la probabilidad de una falsa aceptación, se puede ver el proceso de comparar dos códigos generados por iris distintos como medir la fracción de bits desiguales de cada código, donde cada bit tiene una probabilidad p de ser igual en ambos códigos.

Es posible entonces modelar una única comparación con una distribución binomial con parámetros (N, p), donde N es la cantidad de grados de libertad de cada código de iris. Si bien cada código

tiene un total de $240 \cdot 20 = 4800$ bits, gran parte de los bits de cada código está fuertemente correlacionado con los bits adyacentes.

Como se mencionó antes, se realizan varias comparaciones entre dos códigos, haciendo rotaciones para que el método sea invariante frente a rotaciones. El modelo binomial sirve para modelar cada comparación por separado, por lo que primero se hará un modelo para una única comparación (es decir, sin tener en cuenta las rotaciones) y luego se extenderá el modelo para tener en cuenta todas las rotaciones.

En el modelo binomial para una única comparación, se pueden estimar los valores de p y N haciendo una comparación de todos los códigos de iris distintos y midiendo la distancia de Hamming entre cada código sin tener en cuenta las rotaciones entre los códigos. La distancia de Hamming entre dos códigos se puede ver como una variable aleatoria DH. Luego de comparar todos los códigos, se estima la media μ_{DH} y la varianza σ^2_{DH}. Se calcula el valor de p y N como:

$$p = \mu_{DH} \tag{7.6}$$

$$N = p(1-p)/\sigma^2_{DH} \tag{7.7}$$

La probabilidad de que una fracción $x = m/N$ de los N grados de libertad concuerden en dos códigos de iris distintos se expresa entonces como la función de densidad de la distribución binomial:

$$f(x = m/N) = \binom{N}{m} p^N (1-p)^{N-m} \tag{7.8}$$

Para comprobar que la distribución binomial es un buen modelo, en la figura 7.8 se puede ver el histograma de las distancias de Hamming obtenidas junto con la función de densidad de la distribución binomial con parámetros $p = 0{,}4928$ y $N = 429$, obtenidos a partir de las fórmulas 7.6 y 7.7 en base a un total de 253.178 comparaciones entre códigos de iris distintos. Se puede ver una concordancia perfecta entre el modelo y los resultados obtenidos.

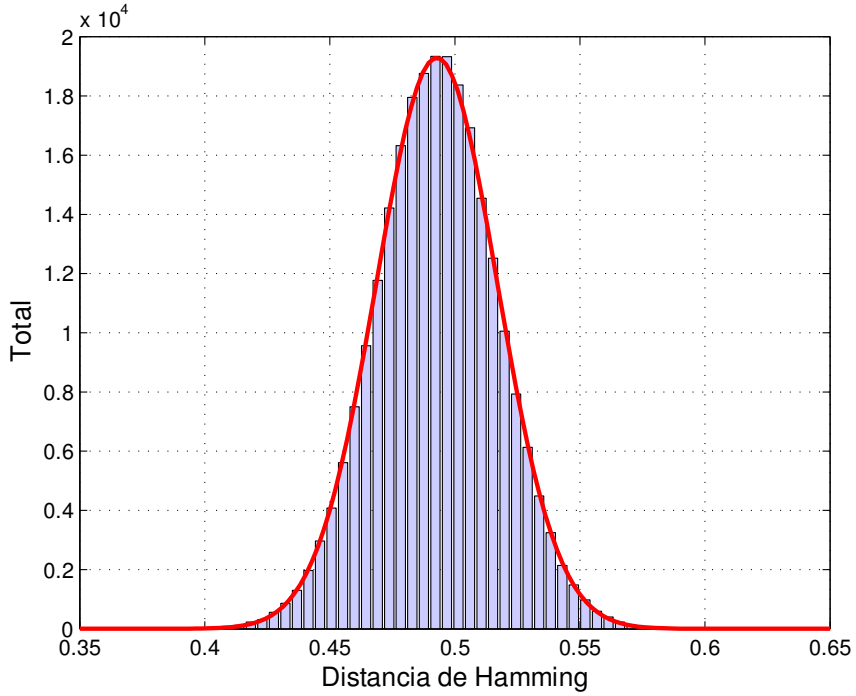


Figura 7.8: Histograma de 253.178 comparaciones inter-clase de códigos de iris. En rojo, la función de densidad de la distribución binomial con parámetros $p = 0{,}4928$ y $N = 429$. Se puede ver una buena concordancia entre el modelo binomial y los resultados obtenidos.

La cercanía del valor de p con 0,5 permite asumir que cada uno de los N grados de libertad tiene dos valores equiprobables, por lo que se puede decir que comparar dos códigos de iris distintos prácticamente equivale a tirar una moneda 429 veces y contar la cantidad de veces que sale *cara*.

Teniendo en cuenta las rotaciones

El problema con el modelo presentado anteriormente es que no tiene en cuenta *todas* las comparaciones hechas entre dos códigos de iris, ya que sólo considera una única comparación. Como se mencionó en el capítulo 5, a la hora de comparar dos códigos de iris, los mismos son rotados para lograr cierta invarianza frente a rotaciones del ojo, y esto introduce algunas modificaciones en el modelo probabilístico.

Daugman introdujo estas modificaciones en el modelo en [Dau03] de la siguiente manera: si $f_0(x)$ es la función de densidad de las distancias de Hamming entre los códigos de iris sin tener en cuenta las rotaciones (ecuación 7.8), entonces la función acumulada de $f_0(x)$, $F_0(x) = \int_0^x f_0(u)\,\mathrm{d}u$ define la probabilidad de una falsa aceptación para el umbral x, como se vio en la figura 7.2. La probabilidad de *no* hacer una falsa aceptación es entonces $1 - F_0(x)$.

Al hacer n rotaciones, se estarán haciendo n comparaciones independientes entre los códigos de iris, por lo que la probabilidad de no hacer una falsa aceptación en ninguna de las comparaciones es $(1 - F_0(x))^n$. Luego, la probabilidad de hacer *al menos una* falsa aceptación en las n pruebas es $F_n(x) = 1 - (1 - F_0(x))^n$. Finalmente, la función de densidad de la distancia de Hamming luego de n rotaciones, $f_n(x)$, es:

$$f_n(x) = \frac{\mathrm{d}}{\mathrm{d}x} F_n(x) = n f_0(x)\,(1 - F_0(x))^n \tag{7.9}$$

Al graficar el histograma de las comparaciones inter-clase con 20 rotaciones (10 a la izquierda y 10 a la derecha, figura 7.9), se obtiene nuevamente una buena concordancia respecto a los resultados previstos por el modelo teórico.

Una vez calculada la función de densidad de las distancias de Hamming, la probabilidad de una falsa aceptación para un theshold x es simplemente la función acumulada de $f_n(x)$, $F_n(x)$, como se definió anteriormente. En el siguiente cuadro se puede ver la probabilidad de falsa aceptación para algunos valores de umbral, calculados a partir de la función acumulada:

Umbral	Probabilidad de falsa aceptación
0.38	1 en 1.730.000
0.37	1 en 16 millones
0.36	1 en 127 millones
0.35	1 en 83.000 millones
0.3	1 en 500.000 billones

Son estas probabilidades extremadamente bajas las que permiten que el iris pueda ser utilizado para la identificación de personas. Por ejemplo, en una base de datos de 100.000 personas (un tamaño considerable), la probabilidad de una identificación errónea de una persona para un umbral de 0,35 es de apenas el 0,008 %.

7.4. Resumen

En esta sección se analizaron los resultados de los algoritmos implementados en el sistema utilizando dos bases de datos de iris disponibles públicamente. Por una parte se analizó el método de segmentación implementado, obteniendo muy buenos resultados con un algoritmo sumamente eficiente y flexible. Por otra parte, se analizaron los resultados del método de codificación,

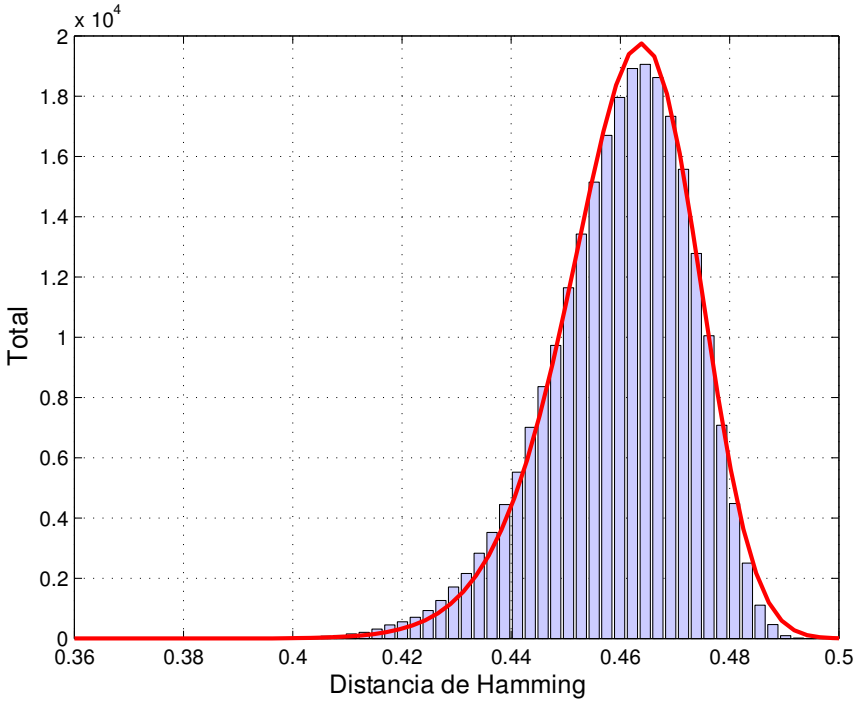


Figura 7.9: Histograma de distancias de Hamming entre códigos de iris con 20 rotaciones. La línea sólida representa la distribución expresada en la ecuación 7.9. Se puede ver nuevamente una concordancia perfecta entre el modelo teórico y los resultados.

obteniendo resultados perfectos para la base de datos CASIA-1 y buenos resultados para la base MMU. Finalmente, se hizo un análisis teórico de la probabilidad del error en la identificación, y se comprobó que los resultados obtenidos encajan perfectamente con la teoría, obteniéndose probabilidades de error sumamente pequeñas.

Capítulo 8

Protección frente a falsificaciones

Una característica muy importante que deben poseer los sistemas de seguridad, y en especial los basados en técnicas biométricas, es la de la protección frente a posibles falsificaciones. En un sistema de seguridad biométrico, una falsificación consiste en presentar al sistema una muestra generada especialmente el fin de engañar al sistema de una de dos formas posibles:

- Hacer que el sistema identifique a una persona como otra (es decir, generar una *falsa aceptación*)
- Hacer que el sistema *no* reconozca a la persona como quien realmente es (generar un *falso rechazo*)

El primer caso es el más común y se da en los casos en los que un usuario no autorizado desea acceder a algún recurso al que solo un grupo de personas registradas en una base de datos tiene acceso.

El segundo caso se puede dar en la situación inversa: por ejemplo, cuando existe una base de datos de personas a las que *no* se le permite el acceso a un determinado recurso. Por ejemplo, en el sistema de reconocimiento de iris instalado en los aeropuertos de los Emiratos Árabes Unidos se posee una base de datos de personas deportadas a las que no se les permite reingresar al país [DM04]. En este caso, una falsificación consistiría en presentar al sistema un iris "falso" (mediante, por ejemplo, un lente de contacto con una textura de iris impresa) para que el sistema no pueda identificar a la persona.

Para evitar estas situaciones, el sistema debe ser capaz de detectar cuándo se le está tratando de engañar mediante una muestra falsa. En un sistema biométrico basado en imágenes (como pueden ser los sistemas de identificación de huella dactilar, iris y reconocimiento facial) existen dos formas de presentar una muestra falsa:

- Presentar al sistema simplemente una copia de la muestra biométrica en cuestión[1].
- Modificar una muestra biométrica "viviente" para que se asemeje a otra muestra.

Dependiendo del sistema, estos dos casos pueden ser sencillos o difíciles de detectar. El primer caso se soluciona por lo general con información extra que permita decidir si la muestra biométrica que se le está presentando al sistema proviene de una persona viviente en vez de una impresión u otro tipo de soporte. Para el segundo caso es necesario analizar la muestra biométrica para detectar posibles alteraciones.

Quizás, uno de los mejores ejemplos de falsificación de una muestra biométrica es el presentado en el programa de televisión *Mythbusters* [Myt]: en el programa, se puso a prueba un sistema

[1] *Muestra biométrica* se refiere a la característica biométrica con la que funciona el sistema. Por ejemplo, en el caso de un sistema de identificación de huella dactilar, una muestra biométrica es una huella dactilar particular.

de identificación de huella dactilar comercial con el fin de engañarlo para tener acceso a una puerta protegida. El sistema fue capaz de reconocer cuando se le estaba presentando una fotocopia de una huella digital, ya que el sistema (supuestamente) detecta el pulso, temperatura corporal y transpiración de la persona para evitar este tipo de falsificaciones. Sin embargo, el sistema fue engañado de tres formas distintas:

1. Usando un pedazo de látex húmedo adherido al pulgar
2. Usando una huella impresa en un pedazo de gel balístico
3. Usando una fotocopia de una huella en papel *húmedo* para simular transpiración

En estos tres casos se puede ver que el sistema falla en detectar los dos tipos de falsificaciones. Cabe preguntarse entonces, ¿cómo puede hacer un sistema de reconocimiento de iris para evitar estas falsificaciones?

8.1. Falsificaciones en un sistema de reconocimiento de iris

A diferencia del sistema de reconocimiento por huella dactilar, un sistema de reconocimiento de iris nunca está en contacto directo con la persona, por lo que el problema de detectar si la muestra es real o no en tiempo real (es decir, si proviene de una persona viviente) es más complicado.

En un sistema de reconocimiento de iris, la única muestra biométrica es el ojo del usuario. Es necesario entonces ver de qué manera se puede analizar el ojo presentado al sistema para verificar que es una muestra de una persona viviente y no de una muestra impresa u otra falsificación.

Wildes [Wil97] fue el primero en sugerir la idea de verificar el movimiento de la pupila como factor para decidir si se está evaluando un "espécimen vivo". La pupila se contrae y dilata involuntariamente frente a cambios de iluminación (*reflejo pupilar*), e inclusive tiene pequeñas oscilaciones bajo iluminación constante (*hipo pupilar*). Cuando la intensidad de la luz disminuye, se produce la *midriasis*, que es la dilatación de la pupila con el fin de capturar más luz. En forma inversa, cuando la intensidad de la luz aumenta, se produce la contracción de la pupila, denomidada *miosis*.

Lee [LPK06a] propone detectar los reflejos conocidos como "reflejos de Purkinje", que son los reflejos especulares producidos dentro del ojo (en la parte interna y externa de la córnea y el cristalino)

También propone, en otro trabajo [LPK06b], detectar los cambios en la reflectancia entre el iris y la esclera frente distintas longitudes de onda de luz infrarroja. Bajo los cambios de iluminación, las propiedades de reflectancia de los iris y escleras falsos y verdaderos son notablemente diferente, lo que permite realizar una verificación rápida de la veracidad del iris.

8.2. Solución propuesta

En este trabajo se realizó un estudio de factibilidad de la solución sugerida (pero no implementada) por Wildes [Wil97], es decir, detectar el reflejo pupilar frente a cambios de iluminación para comprobar la veracidad del ojo presentado al sistema. Para esto, es necesario analizar el cambio en el diámetro de la pupila frente a cambios de iluminación.

Para realizar las pruebas, se cuenta con un ojo "falso" impreso en una hoja de papel. Se programó una aplicación utilizando la librería IrisLib encargada de medir el radio de la pupila. El mismo fue normalizado en función del radio del iris, por lo que el radio de la pupila es expresado como un número entre 0 y 1.

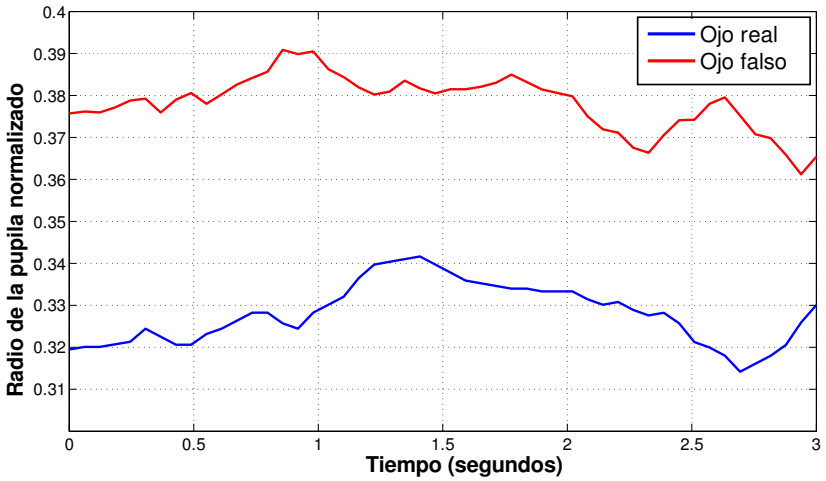


Figura 8.1: Variación del radio de la pupila normalizado para un ojo falso y un ojo verdadero. En el caso del ojo falso, se varió la distancia relativa a la cámara para simular una contracción/dilatación en la pupila.

Se mide entonces, para un lapso de tiempo de 3 segundos, la respuesta del ojo falso y de un ojo real frente a una iluminación constante. El resultado de esto se puede ver en la figura 8.1.

Se puede ver que en ambos casos se obtuvo una variación de aproximadamente el 3 % en el radio de la pupila. En el caso del ojo real, esta variación se produjo por el hipo pupilar descripto anteriormente, mientras que en el caso del ojo falso la variación se obtuvo variando levemente la distancia entre la impresión y la cámara. A simple vista, no existe una diferencia apreciable entre los dos casos, por lo que detectar únicamente el hipo pupilar, como propuso Daugman, no resulta ser una solución viable.

Por lo tanto, se hace un experimento para detectar el *reflejo* pupilar en vez del *hipo* pupilar. Cuando el sistema detecta que hay un ojo presente, emite por la pantalla un destello de corta duración con el fin de provocar un reflejo en la pupila [2]. En la figura 8.2 se puede ver el resultado de este experimento para el mismo ojo.

En este caso, se puede ver cómo la pupila reacciona frente a los flashes de luz, primero contrayéndose y luego dilatándose con la misma frecuencia que los flashes de luz (aunque no simultáneamente ya que el reflejo tarda unas fracciones de segundo en dispararse), mientras que en el ojo falso prácticamente no se producen variaciones en el radio.

De esta forma, detectando la frecuencia de las varaciones en el radio de la pupila y comparándola con la frecuencia de los flashes de luz, es posible verificar la veracidad del ojo.

8.2.1. Detección de lentes de contacto

Otro tipo de problema a solucionar es la presencia de lentes de contacto en el ojo de la persona. El procedimiento descripto anteriormente sirve para detectar los casos en los que un ojo falso es presentado al sistema, sin embargo, en el caso de los lentes de contacto el ojo será reconocido como válido, pero la textura del iris puede ser modificada por el lente. El sistema debe ser capaz entonces de reconocer cuando el usuario esté usando un lente de contacto.

Existen comercialmente tres tipos de lente de contacto:

[2] Esto es muy similar a la funcionalidad de "reducción de ojos rojos" de algunas cámaras fotográficas. Cuando se saca una foto, la cámara dispara dos flashes de luz sucesivos con el fin de contraer la pupila y minimizar el efecto de los ojos rojos provocados por el rebote del flash en la retina.

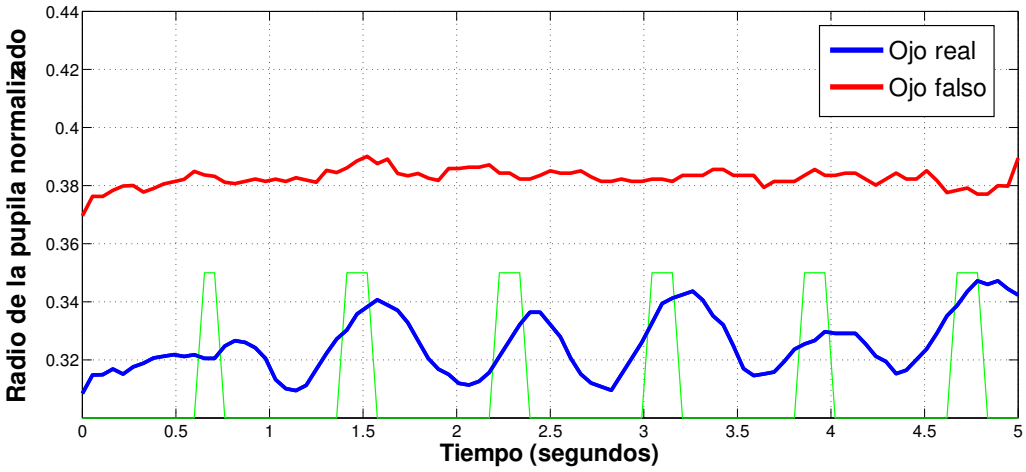


Figura 8.2: Variación del radio de la pupila normalizado para un ojo falso y uno verdadero frente a destellos (flashes) de luz. El momento en que se activan los flashes de luz y su duración aparecen en verde en la parte inferior de la figura. Se puede ver claramente cómo reacciona la pupila verdadera, mientras que no se produce reacción alguna en el ojo falso

1. Lentes de contacto simples
2. Lentes de contacto que cambian el color de ojos
3. Lentes de contacto que realzan o cambian el color de ojos (en inglés, *color-enhancer contact lens*)

De estos tres, el primero no causa problemas ya que son lentes de contacto transparentes sin ningún tipo de impresión. Los otros dos tipos de lentes de contacto poseen impresiones con el fin de cambiar la apariencia del color de ojos de la persona, y dichas impresiones se superponen a la textura del iris.

En la figura 8.3 se pueden ver varios ejemplos de imágenes de ojos con lentes de contacto capturadas bajo luz normal y luz infrarroja. Es posible ver que los lentes de contactos para mejora de color de ojos no causan mayores problemas ya que la textura es prácticamente invisible en el infrarrojo. Sin embargo, los lentes de contacto de color tienen una textura que se ve claramente en el infrarrojo.

von Seelen [vS] propone dos soluciones a estos problemas:

- Registrar los patrones de todos los lentes de contacto en la base de datos y, como primer paso del algoritmo de búsqueda, matchear los iris contra esta lista.
- Realizar un estudio estadístico de la textura de los iris naturales contra los iris impresos para detectar desviaciones que permitan diferenciar uno de otros.

Ambos casos tienen ventajas y desventajas: el primer caso es sencillo de implementar ya que solamente se debe tener una base de datos separada con los posibles patrones de lentes de contacto, pero cualquier patrón nuevo que no exista en la base dará como resultado un falso negativo. El segundo método es más robusto, pero es más proclive a generar falsas alarmas (Seelen reporta un EER de 12 %, que dependiendo de la aplicación puede ser un valor elevado).

8.3. Resumen

En este capítulo se analizaron las distintas formas en los cuales un sistema de reconocimiento de iris puede ser engañado con el fin de generar un falso rechazo o una falsa aceptación. Se diseño

un método que permitiría detectar con relativa facilidad si el ojo que se le presenta al sistema es un ojo "viviente" o no, evitando las posibles falsificaciones. Queda abierto el problema de detectar en forma concluyente si un ojo tiene un lente de contacto que modifique la textura del iris o no.

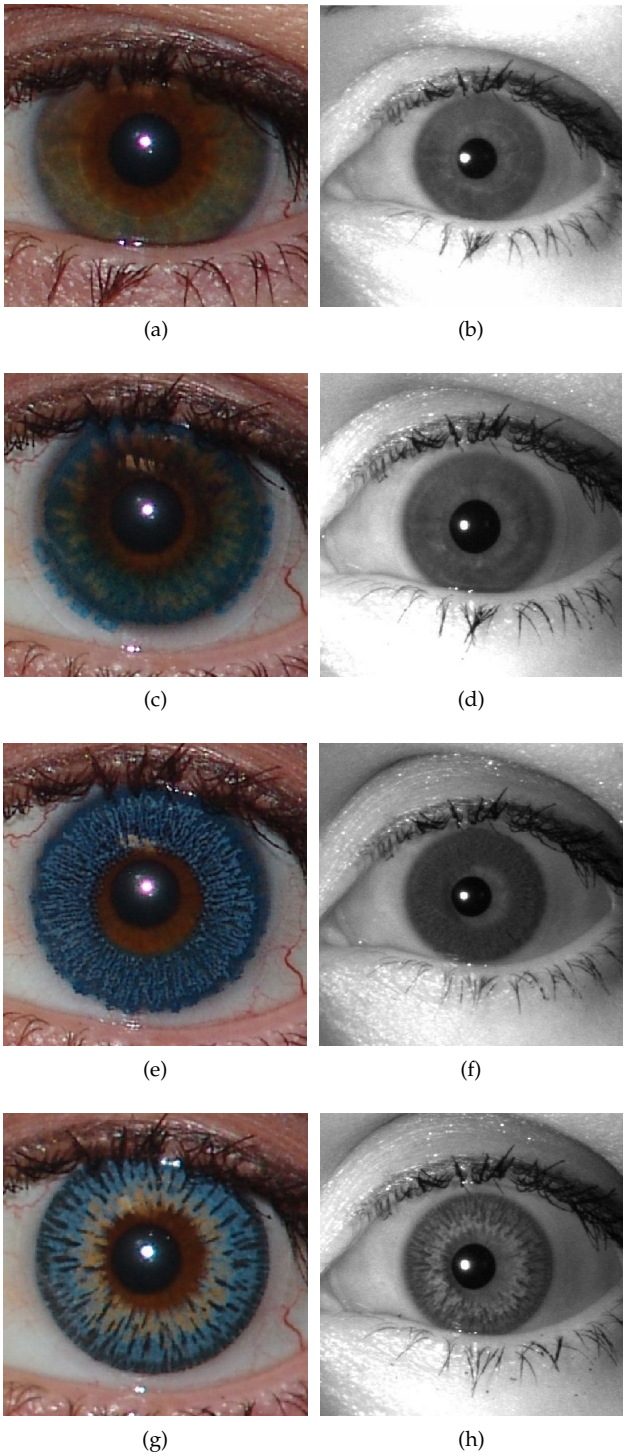


Figura 8.3: Ejemplos de ojos con lentes de contacto con luz visible (columna izquierda) e infrarrojo (columna derecha). 8.3a y 8.3b: Ojo sin lentes de contacto. 8.3c y 8.3d: Ojo con lente de contacto para mejora de color. 8.3e, 8.3f, 8.3g y 8.3h: Ojos con lentes de contacto para cambiar el color de ojo

Capítulo 9

Conclusiones y trabajo futuro

En este trabajo se implementó un sistema de reconocimiento de iris completo, utilizando componentes de hardware y software. Se desarrollaron nuevos algoritmos que permiten el funcionamiento del sistema en tiempo real, con una interacción mínima con el usuario.

El principal aporte de este trabajo es la implementación de una librería que utiliza algoritmos desarrollados especialmente y que están al nivel del estado del arte, junto con un sistema completo que utiliza dicha librería. Además, se hicieron aportes en varias etapas del proceso de reconocimiento de iris:

- En la **captura** se diseñó un sistema que permite capturar imágenes del iris con suficiente resolución para la identificación. Se estableció que ubicando el iluminador infrarrojo en la parte inferior se consigue un mejor contraste en la textura del iris.
- En la **segmentación**, se crearon algoritmos sumamente eficientes y optimizados para funcionar en tiempo real. Se modificó el operador integro-diferencial para que utilice únicamente aritmética entera mediante el algoritmo de Bresenham, lo que optimizó su tiempo de ejecución. Se mejoró el modelo circular clásico para la pupila y el iris reemplazándolo por un modelo basado en contornos. Esto demostró una mejora significativa en la identificación, como se vió en el capítulo 7. Los algoritmos también fueron diseñados para ser flexibles bajo distintas condiciones de iluminación y ruido.
- En el **procesamiento de video** se creó un algoritmo que permite establecer de forma sencilla y rápida la calidad de una imagen, basado en medir la calidad del contorno entre la pupila y el iris. La ventaja de este método es que está diseñado para tener en cuenta la estructura particular del ojo humano. Se definieron heurísticas que permiten detectar si una imagen tiene o no un iris. Juntando estas dos cosas, se definió un método que intenta extraer la mejor imagen de la secuencia de video para la identificación.
- En la **codificación** se buscaron los parámetros óptimos del método de codificación por medios de filtros de Log-Gabor. Combinando la codificación por filtros de Log-Gabor con las mejoras en el algoritmo de segmentación y la utilización de partes específicas del iris, se lograron mejoras significativa frente a los resultados obtenidos en otros trabajos.
- En cuanto a la **protección frente a falsificaciones** se analizó la factibilidad de detectar el reflejo pupilar como medida para evitar fraudes, con resultados positivos.

Los resultados obtenidos fueron sumamente alentadores, obteniendo tasas de error bajísimas en la identificación, lo que permite concluir que el sistema puede ser utilizado en entornos reales.

9.1. Trabajo futuro

El principal problema de las pruebas realizadas radica en que no se poseen suficientes imágenes producidas por el sistema de captura implementado como para poder realizar estadísticas significativas. Es necesario entonces generar una base de datos propia, con suficientes imágenes, y probar el algoritmo de codificación sobre dicha base. Se demostró en la comparación entre la base de datos CASIA-1 y MMU que la calidad de las imágenes impacta fuertemente en el resultado final del sistema, por lo que es necesario analizar la capacidad del sistema de captura de generar imágenes con calidad suficiente como para tener buenos resultados.

Otro trabajo a realizar consiste en relajar un poco las condiciones de captura. En el sistema implementado se requiere que el usuario esté a una distancia bastante corta del lente y que además esté alineado con la cámara. El primer problema puede mejorarse utilizando lentes de mayor longitud focal, que al momento de hacer este trabajo no estaban disponibles fácilmente en el mercado. El segundo problema fue analizado en forma muy reciente por algunos autores [SSA+07, Dau07], y proponen detectar desviaciones del modelo circular del iris y la pupila para normalizar imágenes capturadas en forma no alineada con la cámara. Sería deseable entonces agregar esta funcionalidad en la librería implementada.

Es deseable también realizar más pruebas sobre el mecanismo de protección frente a falsificaciones que se describió en el capítulo 8. Este mecanismo no fue implementado en la versión final de la librería.

Finalmente, sería interesante armar un sistema embebido que utilice la librería implementada, con el fin de implementar una solución portátil de reconocimiento de iris. Esto es posible ya que dicha librería está diseñada para funcionar de manera eficiente, aún con velocidades bajas de procesador.

Apéndice A

Librería *IrisLib*

La librería IrisLib es el componente principal del sistema implementado, y contiene todos los algoritmos descriptos en este trabajo. Tiene una interfaz sumamente sencilla, diseñada para poder realizar pruebas con la mínima cantidad de código posible. La librería fue diseñada en forma modular, por lo que es sumamente sencillo modificar o implementar funcionalidades y algoritmos nuevos.

A continuación, se presenta el listado de funciones de la librería y un pequeño programa de ejemplo.

A.1. API de la librería

La librería tiene un conjunto reducido de clases que se encargan principalmente del procesamiento de video, la segmentación y la codificación. Las principales clases y sus métodos son los siguientes:

- Clase PROCESADORVIDEO: se encarga de procesar el stream de video y de decidir cuándo se capturó una imagen con suficiente calidad. Funciona como una "caja negra", que procesa los frames individuales y en algún momento decide que hay una imagen buena. Sus métodos son los siguientes:
  - PROCESARFRAME(FRAME): procesa un frame de video
  - ULTIMOFRAME(): devuelve información variada sobre el último frame procesado, como la calidad de la imagen, el resultado de la segmentación y el resultado de las heurísticas para verificar si en la imagen hay un iris o no.
  - HAYFRAMEBUENO(): indica si en el buffer hay una imagen lo suficientemente buena para el procesamiento.
  - MEJORFRAME(): devuelve la información sobre la mejor imagen disponible. Ésta es la imagen que debe ser procesada.
- Clase SEGMENTADOR: su funcionalidad consiste en segmentar la pupila y el iris en las imágenes. Posee un único método, SEGMENTAR(IMAGEN), que devuelve el resultado de la segmentación para dicha imagen
- Clase SEGMENTADORPARPADOS: segmenta los párpados en la imagen. Posee un único método, SEGMENTARPARPADOS(IMAGEN, RESULTADO DE SEGMENTACIÓN).

- Clase NORMALIZADOR: genera la imagen normalizada en coordenadas polares para su posterior codificación, como se explicó en el capítulo 5. Posee un único método, NORMALIZAR(IMAGEN, RESULTADO DE SEGMENTACIÓN).
- Clase CODIFICADOR: se encarga de codificar la textura del iris. Posee dos métodos:
  - CODIFICAR(IMAGEN NORMALIZADA, MÁSCARA NORMALIZADA): genera el código de iris para la imagen previamente normalizada.
  - NORMALIZARYCODIFICAR(IMAGEN, RESULTADO DE SEGMENTACIÓN): igual que la anterior, pero normaliza la imagen automáticamente.

Como se dijo antes, la librería posee una interfaz para el lenguaje Python llamada **PyIrisLib** que permite realizar prototipos de programas en pocas líneas de código.

El funcionamiento de la librería es sencillo. Se debe proveer el stream de video procedente del sistema de captura a la clase PROCESADORVIDEO hasta que dicha clase indique que se capturó una imagen suficientemente buena del iris. Dicha imagen debe ser codificada con la clase CODIFICADOR, y el código resultante debe ser buscado en la base de datos. En la figura A.1 se puede ver el diagrama de flujos de una aplicación típica que utiliza esta librería.

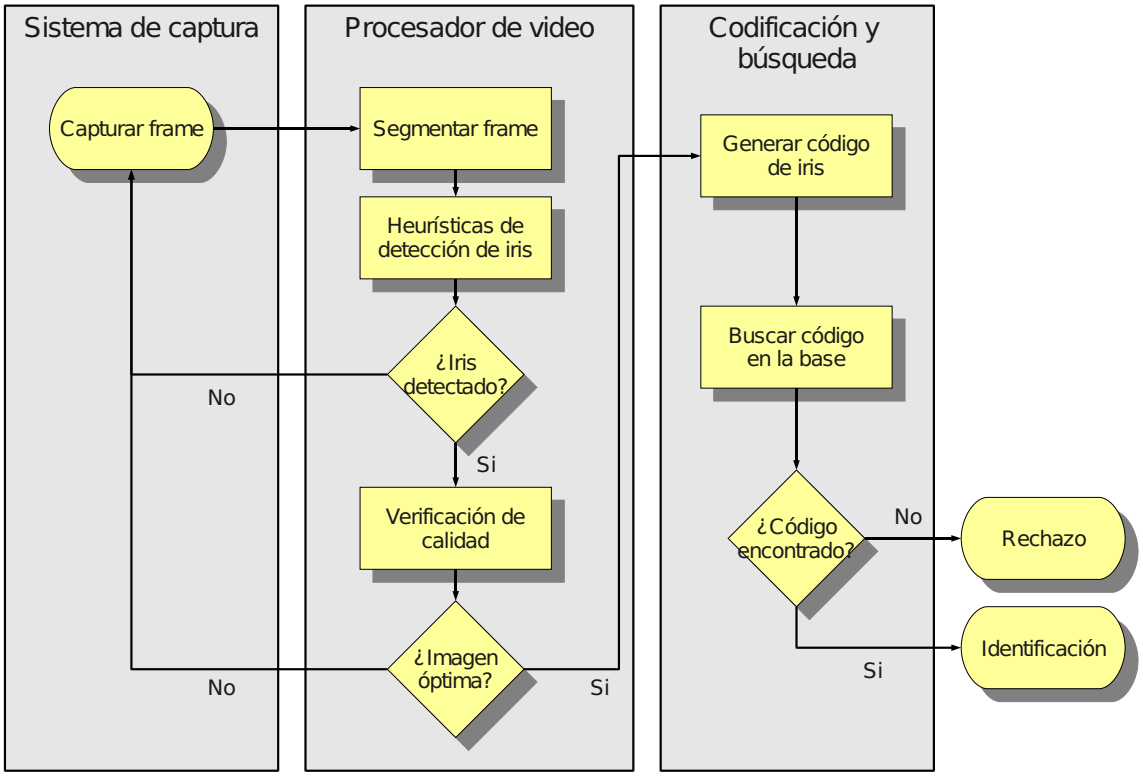


Figura A.1: Diagrama de flujos para un sistema de reconocimiento de iris típico que utiliza la librería IrisLib. La mayor parte de la funcionalidad está encapsulada en unas pocas clases.

A.2. Programa de ejemplo

El siguente es un programa de ejemplo que utiliza la librería. El programa está hecho en el lenguaje Python y, pese a su simplicidad, tiene todas las funcionalidades de un sistema típico de reconocimiento de iris.

Programa de ejemplo

```
# -*- coding: UTF8 -*-
#!/usr/bin/python
# Nota: se eliminan acentos por problemas de compatibilidad
import sys
from opencv import *
from opencv.highgui import *
sys.path.append('..')
import PyIrisLib
import PyIrisLib.BaseIris

# Inicializa el sistema de captura
cap = cvCreateCameraCapture(0)

# Clases de IrisLib
procesadorVideo = PyIrisLib.ProcesadorVideo()
codificador = PyIrisLib.Codificador()
baseIris = PyIrisLib.BaseIris.BaseIris()

# Inicializa la base de datos
baseIris.inicializarBase('base')

# Muestra la ventana de video y la ventana que contiene la ultima imagen
    capturada
cvNamedWindow("video")
cvNamedWindow("mejor imagen")

while 1:
  # Lee los frames del sistema de captura y los muestra en la pantalla de video
  frame = cvQueryFrame(cap)
  cvShowImage("video", frame)

  # Pasa el frame al procesador de video de la libreria
  procesadorVideo.procesarFrame(frame)

  # Hay una imagen buena?
  if procesadorVideo.hayFrameBueno():
    datosMejorFrame = procesadorVideo.mejorFrame()

    # Copia internamente la mejor imagen y la muestra en la pantalla
    segmentacion = datosMejorFrame.segmentacion.copiar()
    mejorImagen = cvCreateImage(cvGetSize(datosMejorFrame.frame), IPL_DEPTH_8U,
        1)
    cvCopy(datosMejorFrame.frame, mejorImagen)
    cvShowImage("mejor imagen", mejorImagen)

    # Genera el codigo de iris
    codificacion = codificador.normalizarYCodificar(mejorImagen, segmentacion)

    # Busca el codigo en la base de datos
    (id_usuario, distanciaHamming) = baseIris.matchCodigo(codificacion.codigo,
        codificacion.mascaraCodigo)
    if not id_usuario:
      # Iris no encontrado en la base
      print "No se encontro el usuario en la base"
```

```
52      else:
53        # Iris encontrado en la base
54        usuario = baseIris.entrada(id_usuario)
55        print 'Identificado el usuario', usuario['nombre_usuario'], 'con una
              distancia de Hamming de', str(distanciaHamming)
56
57    # Finalizacion del programa
58    if cvWaitKey(10) == 'q': break
```

Bibliografía

[19705] ISO/IEC 19794-6, *Information technology: Biometric data interchange format*, 2005.

[AT06] E. Arvacheh and H. Tizoosh, *Iris segmentation: Detecting pupil, limbus and eyelids*, ICIP (2006), 2453–2456.

[AYL06] A. Adler, R. Youmaran, and S. Loyka, *Towards a measure of biometric information*.

[BB98] W. Boles and B. Boashash, *A human identification technique using images of the iris and wavelet transform*, IEEE Transactions on Signal Processing **46** (1998), 1185–1188.

[BRM+06] C. Boyce, A. Ross, M. Monaco, L. Hornak, and X. Li, *Multispectral iris analysis - a preliminar study*, Conference on Computer Vision and Pattern Recognition Workshop (CVPRW'06) (IEEE, ed.), no. 0-7695-2646-2, 2006.

[CWT+] J. Cui, Y. Wang, T. Tan, L. Ma, and Z. Sun, *A fast and robust iris localization method based on texture segmentation*, Tech. report, Institute of Automation, Chinese Academy of Sciences.

[Dau80] J. Daugman, *Two-dimensional spectral analysis of cortical receptive field profiles*, Vis. Res. **20** (1980), 847–856.

[Dau88] ———, *Complete discrete 2-d gabor transforms by neural networks for image analysis and compression*, IEEE Transactions on Acoustics, Speech and Signal Processing **36** (1988), 1169–1179.

[Dau93] ———, *High confidence visual recognition of persons by a test of statistical independence*, IEEE Transactions on Pattern Analysis and Machine Intelligence **15** (1993), no. 11, 1148–1161.

[Dau00] ———, *Biometric decision landscapes*, Tech. Report 482, Computer Laboratory, University of Cambridge, 2000.

[Dau01] ———, *Statistical richness of visual phase information: Update on recognizing persons by iris patterns*, International Journal of Computer Vision **45** (2001), 25–38.

[Dau03] ———, *The importance of being random: Statistical principles of iris recognition*, Pattern Recognition **36** (2003), 289–291.

[Dau04] ———, *How iris recognition works*, IEEE Transactions on Circuits and Systems for Video Technology **14** (2004), no. 1, 21–30.

[Dau07] ———, *New methods in iris recognition*, IEEE Transactions on Systems, Man and Cybernetics **37** (2007), no. 5, 1167–1175.

[Dem06] T. Dembinsky, *Measurement of the information for identification in iris images*, Master's thesis, University of Ottawa, 2006.

[DM04] J. Daugman and I. Malhas, *Iris recognition border-crossing system in the uae*, Tech. report, International Airport Review, 2004.

[FBH+05] C. Fancourt, L. Bogoni, K. Hanna, Y. Guo, R. Wildes, N. Takahashi, and U. Jain, *Iris recognition at a distance*, Lecture Notes in Computer Science **3546** (2005), 1–13.

[Fie87] D. J. Field, *Relations between the statistics of natural images and the response properties of cortical cells*, Journal of the Optical Society of America (1987), 2379–2394.

[Fol90] van Dam Foley, *Computer graphics: principles and practice*, 2nd edition ed., Addison-Wesley, 1990.

[FS87] L. Flom and A. Safir, *Iris recognition system*, Patente US. no. 4.641.349, 1987.

[HCTW06] Y. He, Y. Cui, T. Tan, and Y. Wang, *Key techniques and methods for imaging iris in focus*, The 18th International Conference on Pattern Recognition (ICPR'06), no. 0-7695-2521-0, 2006.

[HMWT] J. Huang, L. Ma, Y. Wang, and T. Tan, *Iris model based on local orientation description*, Tech. report, National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing, P.R. China.

[HTS06] Z. He, T. Tan, and Z. Sun, *Iris localization via pulling and pushing*, 18th International Conference on Pattern Recognition (2006).

[JHG99] B. Jähne, H. Haußecker, and P. Geißler, *Handbook of computer vision and applications - vol. 2 - signal processing and pattern recognition*, Academic Press, 1999.

[Kov99] P. Kovesi, *Image features from phase congruency*, Videre: Journal of Computer Vision Research **1** (1999), 2–26.

[KP07] B. J. Kang and K. R. Park, *Real-time image restoration for iris recognition systems*, IEEE Transactions on Systems, Man and Cybernetics **37** (2007), no. 6, 1555–1566.

[KZ01] W. Kong and D. Zhang, *Accurate iris segmentation based on novel reflection and eyelash detection model*, Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing (2001), 263–266.

[LE04] P. S. Lee and H. T. Ewe, *Individual recognition based on human iris using fractal dimension approach*, Lecture Notes in Computer Science **3072** (2004), 467–474.

[LLBK01] S. Lim, K. Lee, O. Byeon, and T. Kim, *Efficient iris recognition through improvement of the feature vector and classifier*, ETRI Journal **23** (2001), no. 2, 61–70.

[LM05] P. Lili and X. Mei, *The algorithm of iris image preprocessing*, Tech. report, University of Electronic Science and Technology of China, 2005.

[LPK06a] E. Lee, K. R. Park, and J. Kim, *Fake iris detection by using purkinje image*, Springer LNCS 3832: International Conference on Biometrics, 2006, pp. 397–403.

[LPK06b] S. J. Lee, K. R. Park, and J. Kim, *Robust fake iris detection based on variation of the reflectance ratio between the iris and the sclera*, 2006 Biometrics Symposium (2006).

[Mal91] S. Mallat, *Zero-crossings of a wavelet transform*, IEEE Transactions on Information Theory **37** (1991), no. 4, 1019–1033.

[Mas03] L. Masek, *Recognition of human iris patterns for biometric identification*, Master's thesis, School of Computer Science and Software Engineering, University of Western Australia, 2003.

[MIA05] K. Miyazawa, K. Ito, and T. Aoki, *An efficient iris recognition algorithm using phase-based image matching*, Tech. report, Graduate School of Information Sciences, Tohoku University, 2005.

[MNH+06] J. Matey, O.Ñarodistsky, K. Hanna, R. Kolczynski, D. LoIacono, S. Mangru, M. Tinker, T. Zappia, and W. Zhao, *Iris on the move*, Procedings of the IEEE **94** (2006).

[MRZ07] D. Monro, S. Rakshit, and D. Zhang, *Dct-based iris recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence **29** (2007), no. 4, 586–595.

[MTWZ03] L. Ma, T. Tan, Y. Wang, and D. Zhang, *Personal identification based on iris texture analysis*, IEEE Transactions on Pattern Analysis and Machine Intelligence **25** (2003), no. 12, 1519–1533.

[MTWZ04] L. Ma, T. Tan, Y. Wang, and D. Zhang, *Efficient iris recognition by characterizing key local variations*, IEEE Transactions on Image Processing **13** (2004), no. 6, 739–750.

[Myt] *Mythbusters - episodio nº59 - "crimes and myth-demeanors 2"*.

[OL81] A. V. Oppenheim and J. S. Lim, *The importance of phase in signals*, Proceedings of the IEEE **69** (1981), 529–541.

[RM07] S. Rakshit and D. Monro, *Medical conditions: Effect on iris recognition*, Tech. report, University of Bath, UK, 2007.

[SASR05] C. Sanchez-Avila and R. Sanchez-Reillo, *Two different approaches for iris recognition using gabor filters and multiscale zero-crossing representation*, Pattern Recognition **38** (2005), 231–240.

[SCZY02] E. Sung, X. Chen, J. Zhu, and J. Yang, *Towards non-cooperative iris recognition systems*, Seventh International Conference on Control, Automation, Robotics and Vision (2002), 990–995.

[Sou06] The Imaging Source, *Lenses - selection and setup*, The Imaging Source, Octubre 2006.

[SSA+07] S. Schuckers, N. Schmid, A. Abhyankar, V. Dorairaj, C. Boyce, and L. Hornak, *On techniques for angle compensation in nonideal iris recognition*, IEEE Transactions on Systems, Man and Cybernetics **5** (2007), 1176–1190.

[TCB06] L. D. Terissi, L. Cipollone, and P. Baldino, *Sistema de reconocimiento de iris*, Revista Argentina de Trabajos Estudiantiles **1** (2006), no. 2, 1–6.

[TZ] T. Tan and Sun. Z., *Note on casia-iris v3*.

[VJ01] P. Viola and M. Jones, *Rapid object detection using a boosted cascade of simple features*, 2001, pp. 511–518.

[vS] U. C. von Seelen, *Countermeasures against iris spoofing with contact lenses*, Iridian Technologies.

[WAG+94] R. P. Wildes, J. C. Asmuth, G. L. Green, S. C. Hsu, R. J. Kolczynski, J. R. Matey, and S. E. McBride, *A system for automated iris recognition*.

[Wil97] R. Wildes, *Iris recognition: An emerging biometric technology:*, Proceedings of the IEEE **85** (1997), no. 9, 1348–1363.

[YZL06] W. Yuan, Linm Z., and Xu L., *A novel and fast iris location algorithm based on the structure of human eyes*, Proceedings of the 6th World Congress on Intelligent Control and Automation (2006), 10388–10392.