IMPLEMENTACIÓN SOFTWARE DE ALGORITMOS PARA LA DETECCIÓN DE RASGOS FACIALES UTILIZADOS COMO INDICADORES DE SOMNOLENCIA EN CONDUCTORES USANDO SECUENCIAS DE IMÁGENES BIOMÉTRICAS

RAFAEL BAYONA ACOSTA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
BUCARAMANGA
2007

IMPLEMENTACIÓN SOFTWARE DE ALGORITMOS PARA LA DETECCIÓN DE RASGOS FACIALES UTILIZADOS COMO INDICADORES DE SOMNOLENCIA EN CONDUCTORES USANDO SECUENCIAS DE IMÁGENES BIOMÉTRICAS

RAFAEL BAYONA ACOSTA

Este proyecto se presenta como requisito para optar al título de Ingeniero Electrónico

Director
M.S.E.E. ANA BEATRIZ RAMIREZ

Codirector
ING. LEANDRO FABIO ARIZA

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
BUCARAMANGA
2007

AGRADECIMIENTOS

A Dios por iluminarme y mostrarme cada día el camino a seguir.

A Ana Beatriz Ramírez Silva por ser una gran directora, profesora y amiga.

A mi mama que siempre creyó en mí, a mi papa que me apoyo y animo en todo momento, a mani mi compañero fiel, a Malle mi amor.

A mi tío Carlos por prestarme la cámara digital una y otra vez.

A todos mis primos, tíos y amigos quienes participaron como modelos en la obtención de las fotografías y videos de la base de datos.

A los compañeros del grupo CPS especialmente a mi codirector Leandro Fabio Ariza por sus consejos y opiniones.

A la escuela de ingenierías Eléctrica, Electrónica y Telecomunicaciones y a la Universidad Industrial de Santander.

CONTENIDO

Pág.

LISTA DE FIGURAS

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ANEXOS

Pág.

RESUMEN

TÍTULO: IMPLEMENTACIÓN SOFTWARE DE ALGORITMOS PARA LA DETECCIÓN DE RASGOS FACIALES UTILIZADOS COMO INDICADORES DE SOMNOLENCIA EN CONDUCTORES USANDO SECUENCIAS DE IMÁGENES BIOMÉTRICAS[*].

AUTOR: RAFAEL BAYONA ACOSTA[**]

PALABRAS CLAVES: Procesamiento Digital de Imágenes, segmentación del rostro, detección del iris, compensación de iluminación, correlación y coeficiente de correlación.

DESCRIPCIÓN:

En este trabajo se describe la implementación de un algoritmo que permite, a partir de secuencias de imágenes biométricas de conductores, determinar en tiempo real cambios en las aberturas de los ojos, los cuales pueden ser utilizados para predecir estados de cansancio y somnolencia.

El algoritmo consta de tres etapas. En la primera etapa se realiza una segmentación del rostro la cual se fundamenta en una caracterización universal del color de piel y en el método de crecimiento de regiones por agregación de píxeles. En la segunda etapa, se detecta la localización del iris de uno de los ojos para el caso de personas con los ojos abiertos y se extrae una subimágen de este. Para el caso en que el individuo tenga los ojos cerrados se detecta un punto cualquiera y se extrae una subimágen de esta zona. Para la detección del iris se halla la correlación entre la imágen obtenida con detección de bordes y una máscara con forma de arco. En la tercera etapa para reconocer que la imágen obtenida en la segunda etapa corresponde con la imágen de un ojo abierto, se utiliza el coeficiente de correlación y una imágen patrón. Antes de calcular el coeficiente de correlación se suaviza la imágen con un filtro mediana y se aplica un algoritmo de compensación de iluminación que permita uniformizar los valores de iluminación en la imágen.

La validación de los algoritmos propuestos se hace por medio de una base de datos de imágenes, de personas dentro de un vehículo con diversas condiciones de iluminación, diferentes distancias entre la cámara y la persona y personas en distintas posiciones y con la mirada en diferentes direcciones, adquiridas por medio de una cámara digital, en las cuales se analizan las estadísticas de las frecuencias de falsa aceptación y frecuencias de falso rechazo, la sensibilidad y la especificidad como indicadores de confiabilidad de los algoritmos implementados.

---

[*] Proyecto de Grado

[**] Facultad de Ingenierías Físico-Mecánicas, Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, Director RAMIREZ, Ana Beatriz.

ABSTRACT

TITLE: SOFTWARE IMPLEMENTATION OF ALGORITHMS FOR THE DETECTION OF FACIAL FEATURES USED AS INDICATORS OF DROWSINESS IN DRIVERS USING BIOMETRIC IMAGES SEQUENCES*

AUTHOR: RAFAEL BAYONA ACOSTA**

KEY WORDS: Digital Images Processing, face segmentation, iris detection, illumination compensation, correlation and correlation coefficient.

CONTENT:

This work describes the algorithm implementation that allows starting from biometric images sequences of drivers, to determine changes in the eyes openings in realtime, which can be used to predict fatigue and drowsiness states.

The algorithm consists of three stages. In the first stage a face segmentation is carried out based in a universal skin color map and the region growing method. In the second stage, the iris localization of one eye in the image is detected for people with open eyes and a subimage is extracted. In the case that the individual has the closed eyes anyone point is detected and a subimage of this area is extracted. For the iris detection we calculated the correlation between the edge image and one mask with an arch form. In the third stage it is used the correlation coefficient and a pattern image in order to recognize the previous image as an open eye image. Before calculating the correlation coefficient the image is smoothing with a median filter and an illumination compensation algorithm that is applied to make uniform the illumination values in the image.

The validation of the proposed algorithms was made using a database of images of people inside a vehicle, acquired with a digital camera, in which the statistics of the false acceptance rate and the false rejection rate, the sensibility and the specificity are analyzed as dependability indicators of the implemented algorithms.

---

* Project of Grade

** Ability of Physical-mechanical Engineerings, Electric, Electronic School of Engineerings and Telecommunications, Managing RAMÍREZ, Ana Beatriz

INTRODUCCIÓN

Recientes estudios de las Secretarias de Tránsito, indican que una de las principales causas de muerte en Colombia, son los accidentes de tránsito. Estos accidentes, principalmente en carreteras, generados por cansancio y agotamiento físico de los conductores que deteriora algunas variables funcionales psicomotoras y neurocognoscitivas. Entre estas variables se encuentran el tiempo de reacción, la capacidad de vigilancia, juicio y atención, al igual que el procesamiento de información.

Un método eficiente y no intrusivo que opera en tiempo real con buena precisión y alta confianza, consiste en emplear tecnología electrónica y técnicas para la adquisición y procesamiento de imágenes.

En este trabajo se desarrollan algoritmos de Procesamiento Digital de Imágenes (PDI), particularmente biométricas, para detectar los ojos en una imágen facial y los cambios en las aberturas de los mismos, los cuales pueden ser utilizados para predecir estados de somnolencia y cansancio en conductores.

El presente trabajo forma parte de una las líneas de investigación del grupo de investigación en Conectividad y Procesado de Señal (CPS) de la Escuela de Ingeniería Eléctrica, Electrónica y Telecomunicaciones.

En el primer capítulo se explican los conceptos fundamentales de imágenes biométricas y las técnicas de procesamiento digital de imágenes y secuencias de imágenes.

En el segundo capítulo se expone de forma detallada el algoritmo implementado para la detección de rasgos faciales y características de somnolencia. Este consta

de tres etapas: segmentación del rostro, detección del iris y reconocimiento de imágenes.

En el tercer capítulo se muestran los resultados experimentales obtenidos en la validación del algoritmo implementado con una base de datos de 137 imágenes a color. Las imágenes fueron adquiridas dentro de un vehículo con diversas condiciones de iluminación, diferentes distancias entre la cámara y la persona y personas en distintas posiciones y con la mirada en diferentes direcciones. Se analizaron las estadisticas de las ratas de falso acierto y falso rechazo, la sensibilidad y la especificidad.

Por último se presentan las conclusiones y las recomendaciones respectivas.

1. MARCO TEÓRICO

1.1. IMÁGENES BIOMÉTRICAS

Las imágenes de particular interés en el desarrollo de este proyecto son conocidas como imágenes biométricas y son estudiadas por la *biometría*. La palabra biometría proviene del antiguo griego: *bios* (vida) y *metron* (medida). Por lo tanto, la biometría consiste en el estudio de los métodos de medición de características propias de los seres vivos, en particular seres humanos. Asimismo, un equipo de medición de estas características se conoce como equipo biométrico.

La biometría estudia los métodos de identificación y autenticación de los seres humanos a través de sus características propias fisiológicas y de su comportamiento. Las características fisiológicas más comunes son la geometría de la mano, el iris, la retina, la huella digital y el reconocimiento facial. Por otro lado, algunas de las características de comportamiento usadas en identificación de personas son la firma y la voz.

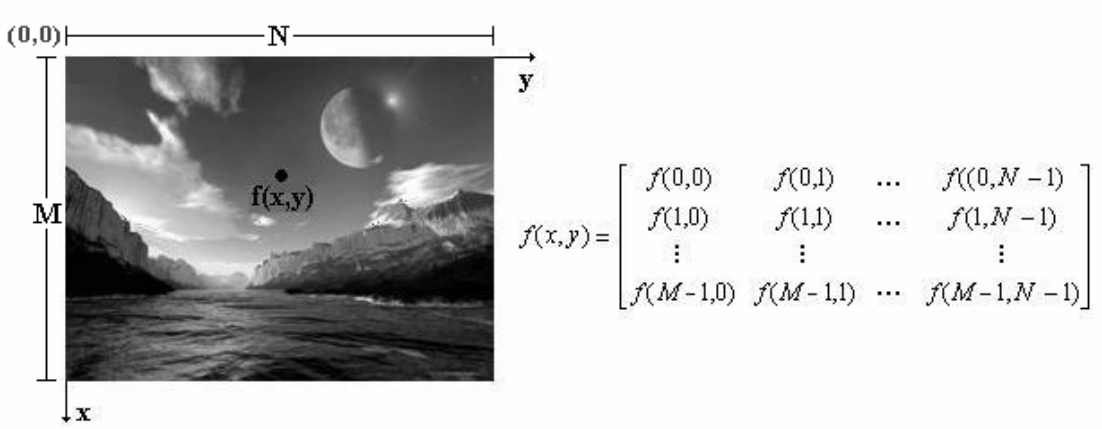
1.2. PROCESAMIENTO DIGITAL DE IMÁGENES

El creciente desarrollo de la electrónica y los sistemas de cómputo ha permitido que las técnicas de procesamiento digital de imágenes sean utilizadas como un método no intrusivo y automático para extraer información de las características físicas presentes en una escena en particular.

1.2.1. Representación de una Imágen Digital. Una imágen se representa por una función bidimensional *f(x,y)*, donde "*x*" e "*y*" son las coordenadas espaciales y el valor de *f* en cualquier punto es proporcional al nivel de intensidad o nivel de gris de la imágen en ese punto. Una imágen digital es una imágen en la cual se

realiza un muestreo de las coordenadas espaciales *(x,y)* y se cuantifica el nivel de intensidad. Matemáticamente, una imágen digital se representa por una matriz de *M* x *N,* donde cada elemento de la matriz es una cantidad discreta y equivale al nivel de intensidad en esa posición. A cada elemento de la matriz, se le denomina, elemento de la imágen o *píxel* (abreviatura de su denominación en ingles, *picture elements*). La resolución o grado de detalle discernible en una imágen depende del tamaño de la imágen *M* x *N* (resolución espacial) y del número de niveles de gris *L* (resolución de amplitud), el cual comúnmente es una potencia de 2. En la figura 1 se muestra una imágen digital y su representación.

Figura 1. Representación de una imágen digital



Fuente: Autor del proyecto

1.2.2. Etapas Fundamentales de un Sistema de Procesamiento Digital de Imágenes. A continuación se describen las etapas fundamentales de un sistema general de procesamiento digital de imágenes, sin embargo puede haber sistemas en los que no sea necesario realizar algunas etapas.

1.2.2.1. Adquisición. Para la adquisición de una imágen digital es necesario un dispositivo físico sensible a una determinada banda del espectro electromagnético (ultravioleta, visible o infrarrojo) y un dispositivo que digitalice la señal de salida análoga del dispositivo sensor que es proporcional a la energía detectada.

1.2.2.2. Preprocesamiento. En esta etapa el principal objetivo es el de mejorar ciertas características de la imágen adquirida como el contraste, la iluminación y eliminación de ruido entre otros, que permitan adecuar la imágen para su procesamiento posterior.
El filtrado es una operación básica del preprocesamiento ya que permite mejorar la imágen, resaltar aspectos deseados y suprimir aspectos no deseados, como el ruido.

1.2.2.3. Segmentación. Este procedimiento consiste en extraer y aislar uno o varios objetos de interés que constituyen una imágen, basándose en sus propiedades de discontinuidad espacial y similitud. Entre los métodos básicos de segmentación tenemos:

- Detección de discontinuidades.
- Umbralización.
- Crecimiento de regiones por agregación de píxeles.
- División y fusión de regiones.
- Morfología watersheds.

En el desarrollo de este proyecto se utilizan los métodos de detección de bordes y crecimiento de regiones por agregación de píxeles los cuales se explican con detalle en el siguiente capítulo.

1.2.2.4. Representación y Descripción. Después de la segmentación es necesario representar el conjunto de píxeles obtenidos de una forma adecuada

para el procesamiento por computadora. La representación se puede realizar ya sea describiendo características de contorno tales como longitud, curvatura, simetría y forma o características de región tales como textura, color, área y propiedades topológicas.

1.2.2.5. Reconocimiento y Clasificación. El reconocimiento es el proceso por el cual se le da una denominación a una imágen basándose en la información proporcionada por sus descriptores o parámetros característicos de igual forma un sistema de clasificación determina a que grupo pertenece una muestra de acuerdo con unos parámetros dados. Existen diversos tipos de sistemas de clasificación de acuerdo con la complejidad de la aplicación y la naturaleza de los descriptores o parámetros característicos. Algunos de los sistemas de clasificación son:

- Clasificador por frontera de decisión.
- Clasificador de mínima distancia.
- Clasificadores estadísticos bayesianos.
- Lógica difusa.
- Redes neuronales.
- Inteligencia artificial.

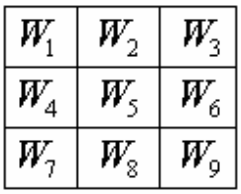
En este proyecto se utiliza un sistema de clasificación simple por frontera de decisión el cual se explica en el siguiente capítulo.

1.2.3. Operaciones Básicas. Algunas operaciones básicas de procesamiento digital de imágenes aplicadas en el desarrollo del algoritmo propuesto se describen a continuación.

1.2.3.1. Filtrado Espacial. Es una operación en la cual los niveles de intensidad de cada píxel en una imágen dependen de los valores de intensidad de los píxeles vecinos en un entorno y de los valores de intensidad de los píxeles de una

subimágen o máscara de igual dimensión que el entorno. Los valores de la máscara son llamados coeficientes o pesos. Esto se ilustra en la figura 2 para una máscara de 3x3.

Figura 2. Una máscara 3x3 con coeficientes arbitrarios



Fuente: Autor del proyecto

Para filtros lineales, el proceso consiste en desplazar la máscara por toda la imágen, centrando las operaciones sobre los píxeles que se encuadran en la región de la imágen original que coincide con la máscara. El resultado del proceso se obtiene mediante una suma de convolución entre los píxeles originales y los diferentes coeficientes de la máscara. En general el filtrado espacial lineal de una imágen *f(x,y)* de tamaño *M* x *N* con una máscara *w(s,t)* de dimensión *m* x *n* está dado por la siguiente expresión:

$$g(x,y) = \sum_{s=-a}^{a}\sum_{t=-b}^{b} w(s,t)\cdot f(x+s,y+t)$$

donde $a=(m-1)/2$, $b=(n-1)/2$, $x=0,1,...,M-1$ y $y=0,1,...,N-1$

Los filtros espaciales no lineales también operan sobre entornos. Sin embargo, su operación se basa directamente en los valores de los píxeles en el entorno en consideración. Unos ejemplos de filtros no lineales habituales son los filtros mínimo, máximo y mediana que son conocidos como filtros de rango ó filtros estadísticos.

1.2.3.2. Filtro Mediana. Es un filtro espacial no lineal, en el cual el nivel de intensidad de cada píxel es reemplazado por la mediana de los niveles de intensidad en un entorno de este píxel. Para realizar este filtrado primero se selecciona el entorno ya sea 3x3, 5x5, 7x7 etc; luego se extraen los valores del píxel y de su entorno, se determina la mediana de estos valores y se asigna este valor al píxel en cuestión.

Este filtro es utilizado para suavizar la imágen y eliminar ruido aleatorio, preservando los bordes.

1.2.3.4. Binarización. Es una transformación no lineal de los niveles de intensidad en una imágen, cuyo resultado es una imágen binaria. En este tipo de imágen cada píxel puede tomar uno de dos posibles valores 0 y 1, donde 0 representa el color negro y 1 representa el color blanco.

1.2.3.5. Umbralización. Consiste en agrupar o clasificar los píxeles de una imágen de acuerdo al valor de intensidad de estos. En el caso mas simple de umbralización se selecciona un valor umbral único de intensidad *T*. Aquellos píxeles de la imágen original *f(x,y*) con valores mayores a este umbral asumirán el valor de 1 (blanco) y los demás píxeles asumirán el valor de 0 (negro) dando como resultado la imágen binaria *g(x,y)*.

$$g(x,y) = \begin{cases} 1 & si \quad f(x,y) > T \\ 0 & si \quad f(x,y) \leq T \end{cases}$$

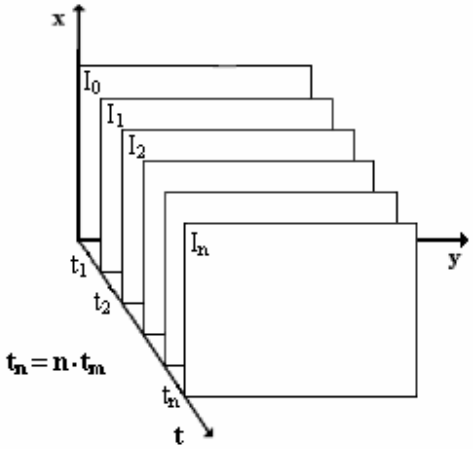
En algunas aplicaciones es útil trabajar con valores normalizados de umbral. El umbral *T* se puede normalizar en el rango de [0,1] dividiendo este valor entre el número de niveles de intensidad *L*.

$$T_n = \frac{T}{L}$$

1.3. SECUENCIAS DE IMÁGENES

Una secuencia de imágenes se obtiene con la adquisición consecutiva y periódica en el tiempo de imágenes en una escena. En una secuencia de imágenes existe una dimensión temporal discreta en la cual sus valores son múltiplos enteros del tiempo de muestreo *tm* como se ilustra en la figura 3.

Figura 3. Representación de una secuencia de imágenes



Fuente: Autor del proyecto

Un parámetro característico en una secuencia de imágenes es el número de cuadros o *frames* por segundo (fps). Algunas cámaras de video digitales disponibles comercialmente tienen valores estándar de 30 y 25 fps. El número de *frames* por segundo fps en una secuencia de imágenes es el inverso del tiempo de muestreo *tm*.

$$fps = \frac{1}{t_m}$$

El tiempo de duración *td* de una secuencia de imágenes en función del número total de cuadros o imágenes *Nf* y el tiempo de muestreo *tm* es:

$$t_d = t_m \cdot (Nf - 1)$$

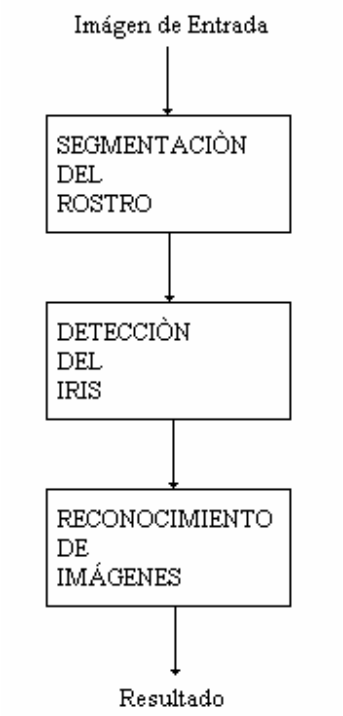
La variación espacio-temporal de la intensidad de la imágen en una secuencia de imágenes nos proporciona información acerca del movimiento presente en la escena, donde la variación espacial se refiere a las dimensiones de la imágen y la variación temporal al eje del tiempo sobre el cual evoluciona la imágen.

2. ALGORITMO PARA LA DETECCIÓN DE RASGOS FACIALES

2.1. GENERALIDADES

El algoritmo usado para el reconocimiento de características de somnolencia consta de tres etapas como se representa en la figura 4. En la primera etapa se realiza una segmentación del rostro, cuya entrada es una imágen de color en el modelo RGB y como resultado se obtiene una imágen en nivel de gris, del área de los ojos en el rostro. En la segunda etapa, se detecta la localización del iris de uno de los ojos para el caso de personas con los ojos abiertos y se extrae una subimágen de este. Para el caso en que el individuo tenga los ojos cerrados se detecta un punto cualquiera y se extrae una subimágen de esta zona. En la tercera etapa se hace un reconocimiento de las imágenes obtenidas en la segunda etapa. El resultado del algoritmo completo determina si la imágen de entrada corresponde a la de una persona con los ojos abiertos o con los ojos cerrados.

Figura 4. Etapas del algoritmo



Fuente: Autor del proyecto

2.2. SEGMENTACIÓN DEL ROSTRO

Este algoritmo se fundamenta en una caracterización universal del color de piel, en el modelo de color YCbCr y en métodos básicos de segmentación, tales como el crecimiento de regiones por agregación de píxeles.

2.2.1. Modelos de Color. En esencia, un modelo de color es una especificación de un sistema de coordenadas tridimensional y un subespacio dentro de dicho sistema donde cada color se representa por un único punto. Las imágenes en un modelo de color, consisten en tres planos de imágen independientes, donde cada plano corresponde con una coordenada.

La mayoría de los modelos de color están orientados bien hacia el hardware como es el caso del modelo RGB, o bien hacia aplicaciones donde se pretende manipular el color como en el modelo YCbCr.

2.2.1.1. El Modelo RGB. En este modelo [5] cada color aparece en sus componentes espectrales primarias: rojo, verde y azul. Está basado en el sistema de coordenadas cartesianas en el cual cada eje corresponde a un componente espectral primario.

Las imágenes del modelo RGB consisten en tres planos de imágen independientes, uno por cada color primario. En un monitor estas tres imágenes se combinan en la pantalla fosforescente para producir una imágen en color compuesta. De forma alternativa la mayoría de las cámaras fotográficas y de video digitales utilizan el formato RGB, razón por la cual este es el formato usado en las imágenes adquiridas.

2.2.1.2. El Modelo YCbCr. En este formato [6] la luminancia es representada por una sola componente, Y, y la información de color es almacenada en dos componentes diferentes, Cb y Cr. La componente Cb es la diferencia entre el color azul y un valor de referencia, y la componente Cr es la diferencia entre el color rojo y un valor de referencia.

Para convertir una imágen del modelo RGB al modelo YCbCr se utiliza la siguiente transformación:

$$\begin{bmatrix} Y \\ Cb \\ Cr \end{bmatrix} = \begin{bmatrix} 16 \\ 128 \\ 128 \end{bmatrix} + \frac{1}{255} \cdot \begin{bmatrix} 65.481 & 128.553 & 24.966 \\ -37.797 & -74.203 & 112 \\ 112 & -93.786 & -18.214 \end{bmatrix} \cdot \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

2.2.2. Caracterización del Color de Piel. Según Douglas Chai y King N. Ngan [1], el color de piel de personas de cualquier raza, en imágenes con condiciones diversas de iluminación, está distribuído en un rango estrecho de las componentes de color Cb y Cr, independientemente de la componente de luminancia Y.

2.2.3. Crecimiento de Regiones por Agregación de Píxeles. Se trata de agrupar píxeles o subregiones en regiones más grandes. La clave de este método consiste en la adición de píxeles, en donde se comienza con uno o varios puntos generadores, a partir de los cuales se van agregando los píxeles contiguos que tengan propiedades similares como nivel de intensidad, color, textura etc.

El desafío principal de este método consiste en la selección apropiada de los puntos generadores que representen correctamente a las regiones de interés y la selección de las propiedades adecuadas para la inclusión de puntos en el proceso de crecimiento.

Para la segmentación del rostro en una imágen, se selecciona como punto de partida o punto generador, un píxel que corresponda a la piel del rostro de la persona y tomando como propiedad niveles de color cercanos al punto generador usando los planos Cb y Cr, de acuerdo con la caracterización del color de piel.

Para considerar que un píxel tiene un nivel de color cercano al nivel de color *Zp* del punto generador, se fija un valor umbral *T* para la diferencia absoluta entre el nivel de color del punto generador y el punto en cuestión. Es decir un píxel con nivel de color *Zi* cumple con la propiedad si:

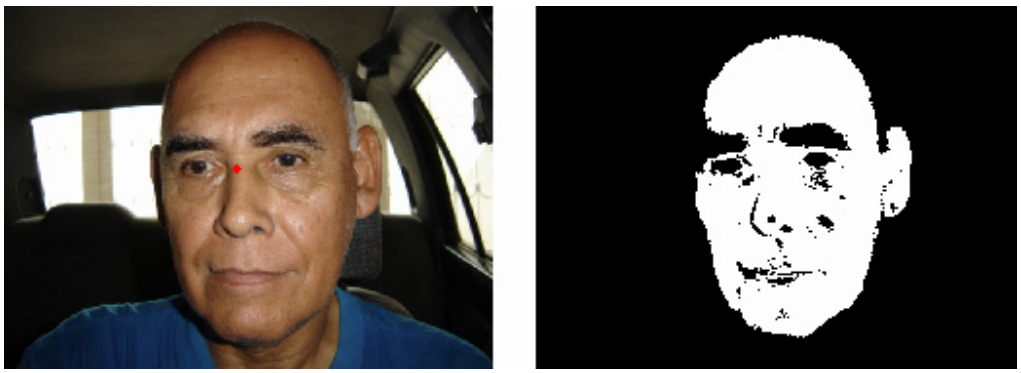
$$|Zi - Zp| < T$$

Si *Zi* y *Zp* tienen las componentes de color *Zicb* y *Zpcb* en el plano Cb, y las componentes *Zicr* y *Zpcr* en el plano Cr entonces:

$$|Zicb - Zpcb| < T \quad \text{y} \quad |Zicr - Zpcr| < T$$

Experimentalmente se encontró que un valor umbral optimo es *T*= 9. En la figura 5 se muestra un ejemplo de la extracción del rostro, usando este umbral y seleccionando como punto generador un punto central. La imágen obtenida es una imágen binaria donde 1 corresponde a los píxeles que cumplen con la propiedad los cuales son agregados al píxel generador.

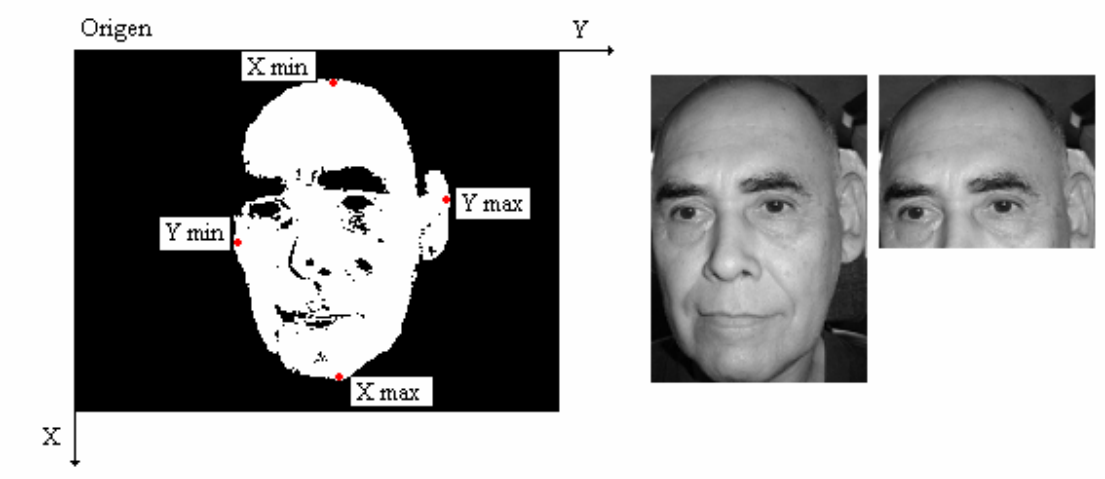
Figura 5. Segmentación del rostro por agregación de píxeles a un punto central en la imágen



Fuente: Autor del proyecto

Determinando las coordenadas máxima y mínima en "*x*" y las coordenadas máxima y mínima en "*y*" de los píxeles blancos en la imágen binaria, se puede extraer una subimágen del rostro. Para el procesamiento posterior a la segmentación no es necesario que la subimágen contenga los tres planos del modelo de color RGB, una imágen de un solo plano en nivel de gris es suficiente. Por lo tanto se toma el plano del color espectral primario Rojo, ya que este se asemeja a la imágen adquirida en nivel de gris. Debido a la ubicación de los ojos en el rostro se puede realizar una segmentación mayor, tomando únicamente la porción superior del rostro. Este proceso se muestra en la figura 6.

Figura 6. Determinación de coordenadas y segmentación de la parte superior del rostro en el plano R



Fuente: Autor del proyecto

Si se selecciona un punto central fijo para segmentar cualquier imágen es posible que este coincida con un punto del rostro que no sea de la piel, como un ojo, una ceja, el cabello o incluso con un punto del fondo, ocasionando una segmentación errónea.

Para determinar el punto generador de manera automática se ubican 4 puntos fijos en la imágen y se selecciona como píxel generador aquel que contiene el mínimo valor de componente Cb de crominancia, el cual corresponde a la piel.

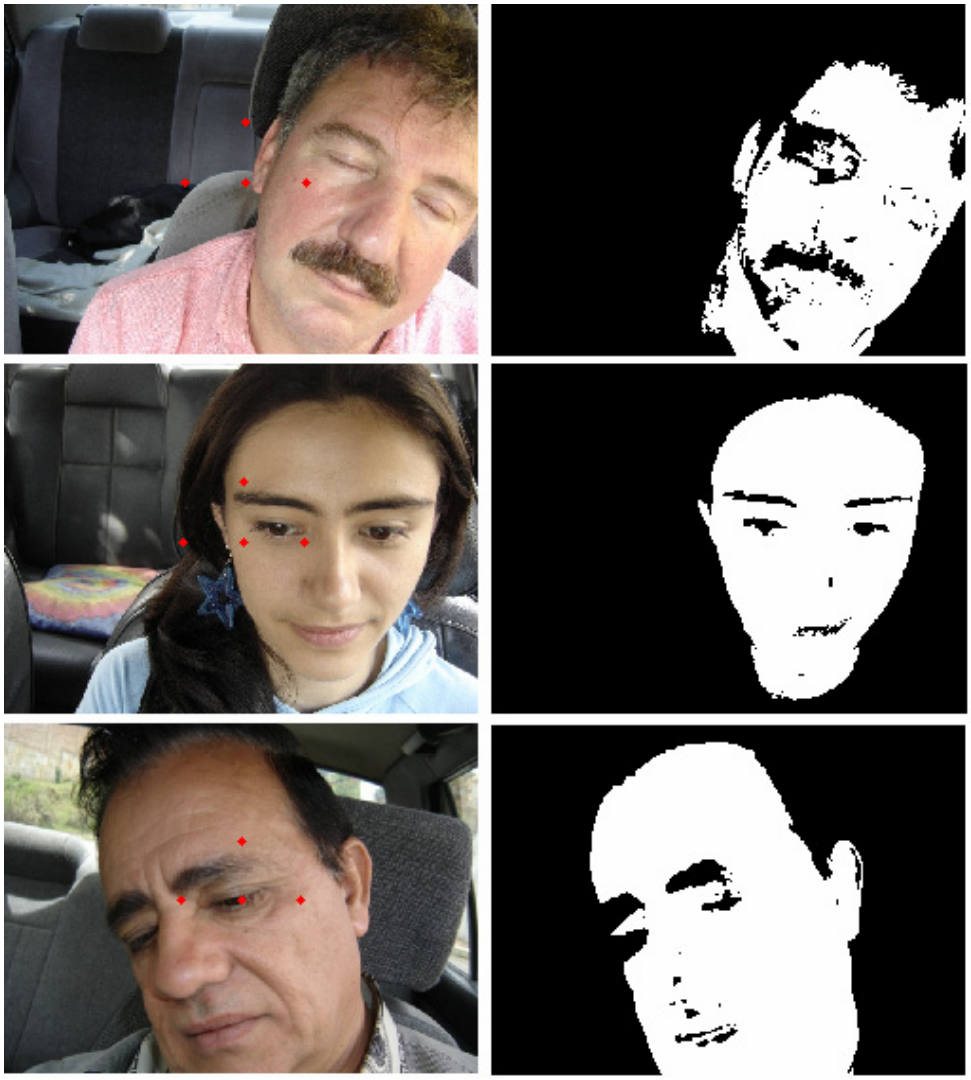
Los puntos están distribuídos de tal forma que se puedan segmentar rostros centrados, inclinados o ubicados a los lados de la imágen. En la figura 7 se muestra la distribución de los puntos utilizando casos de rostros con diferente posición dentro de la imágen.

El algoritmo no requiere que las imágenes iniciales tengan una alta resolución, sin embargo esta característica si es necesaria en la imágen de la zona segmentada.

Por lo tanto para mejorar el rendimiento en el cómputo de este algoritmo se proponen los siguientes pasos:

- Hacer una copia de la imágen de entrada y reducir su tamaño en un factor de escala K.
- Determinar las coordenadas en la versión reducida de la imágen original.
- Multiplicar las coordenadas, por el factor de escala K.
- Extraer la región de interés en la imágen original.

Figura 7. Distribución de los puntos para seleccionar el punto generador



Fuente: Autor del proyecto

2.3. DETECCIÓN DEL IRIS

Este algoritmo se fundamenta en que el nivel de intensidad del iris en una imágen siempre será mayor que el de la esclera, lo cual permite detectar los bordes de forma adecuada sin importar el color de este. Debido a la forma circular del iris se puede encontrar su localización, hallando la correlación entre la imágen obtenida con detección de bordes y una máscara con forma circular según [2].

2.3.1. Detección de Bordes. Un borde en una imágen se define como un cambio en el nivel de intensidad, por lo cual se utilizan operadores diferenciales para su detección. El método más común en procesamiento de imágenes es el vector gradiente. Para una función *f(x,y)*, el gradiente en el punto de coordenadas *(x,y)* se define como:

$$\nabla f = \begin{bmatrix} Gx \\ Gy \end{bmatrix} = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix}$$

La magnitud de este vector es:

$$mag(\nabla f) = \sqrt{(Gx)^2 + (Gy)^2}$$

Para simplificar los cálculos, la magnitud es aproximada utilizando valores absolutos:

$$mag(\nabla f) \approx |Gx| + |Gy|$$

Para calcular las derivadas parciales *Gx* y *Gy* en forma digital se utilizan los operadores *Sobel* que se muestran en la figura 8.

Figura 8. Máscaras para calcular *Gx* y *Gy*

-1	-2	-1
0	0	0
1	2	1

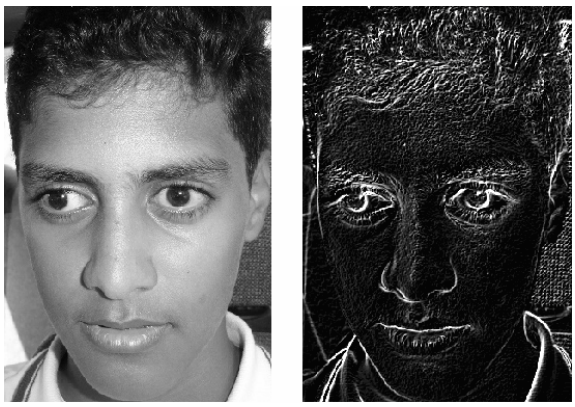
-1	0	1
-2	0	2
-1	0	1

Fuente: Autor del proyecto

Las regiones de intensidad constante en la imágen original aparecen como regiones de intensidad nula en la imágen gradiente. En este caso, los bordes en la imágen gradiente aparecen con niveles de intensidad altos proporcionales a los cambios de intensidad en la imágen original.

Para manipular la imágen gradiente y simplificar los cálculos posteriores, es útil binarizar esta imágen. Ya que los bordes son niveles de intensidad altos y el fondo niveles de intensidad bajos, se utilizo un valor de umbral normalizado de 0.5. Una imágen resultado de este proceso se muestra en la figura 9.

Figura 9. Detección de bordes



Fuente: Autor del proyecto

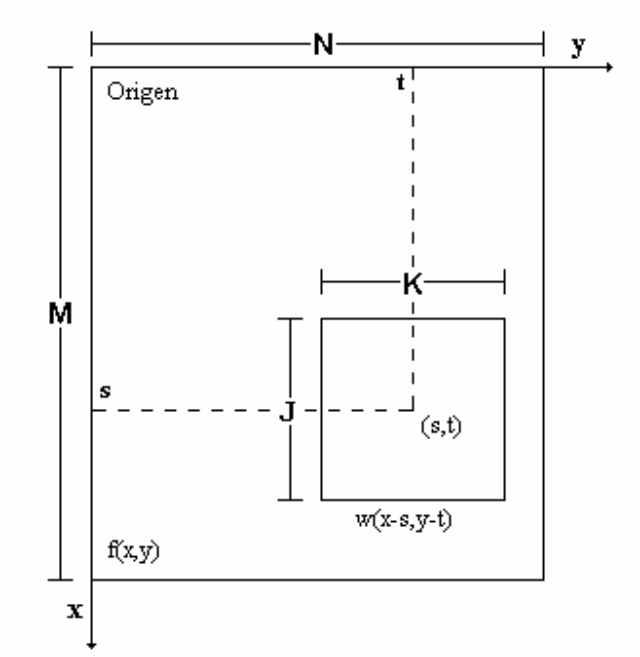
2.3.2. Correspondencia de Imágenes por Correlación. Es la operación que nos permite encontrar réplicas de una subimágen *w(x,y)* de tamaño *J* x *K* dentro de una imágen *f(x,y)* de dimensión *M* x *N*, donde $J \leq M$ y $K \leq N$.
La correlación se define [4] como:

$$c(s,t) = \sum_x \sum_y f(x,y) \cdot w(x-s, y-t)$$

Donde $s = 0,1,2,...,M-1$ y $t = 0,1,2,...,N-1$. El origen de *f(x,y)* está situado en su parte superior izquierda, el origen de *w(x,y)* está ubicado en su centro y las sumatorias se calculan para la región de la imágen donde se solapan *f* y *w*. Al variar *s* y *t*, la subimágen *w(x,y)* se desplazará por la superficie de la imágen *f(x,y)*, dando como resultado la imágen de correlación *c(s,t)*. El máximo valor de *c(s,t)* indica la posición en la que se produce la mayor correspondencia ó similitud entre *w(x,y)* y *f(x,y)*.

En la siguiente figura se ilustra el proceso:

Figura 10. Esquema de correlación



Fuente: Autor del proyecto

Para el diseño de una máscara adecuada para localizar el iris, por medio de correlación se tuvo en cuenta los siguientes aspectos:

- El tamaño del iris presenta pequeñas variaciones entre diferentes personas.
- El tamaño del iris en una imágen depende de la distancia entre la cámara y el individuo. En el caso de un conductor esta distancia puede variar en un rango reducido.
- En personas con ojos parcialmente abiertos solo es visible una porción inferior del iris.
- Para que la imágen de correlación sea prácticamente nula en regiones de intensidad constante de la imágen, la suma de los coeficientes de la máscara debe ser cero ó cercana a cero.

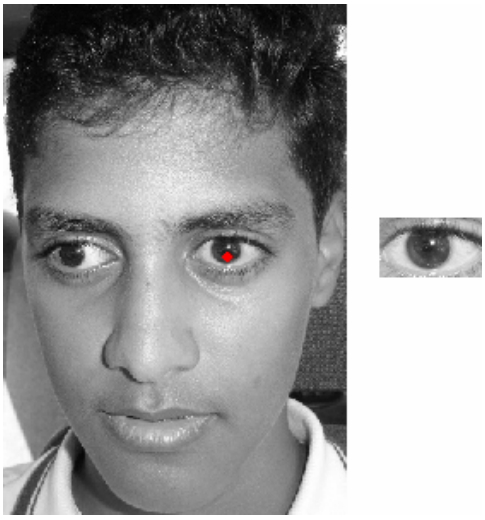
Con base en las consideraciones anteriores se implementó una máscara de 17x31 píxeles, con un radio interno y un radio externo de 11 y 13 píxeles respectivamente. Los coeficientes del fondo equivalen a -1 y los coeficientes del arco a 3, dando una suma total de -11. La imágen de la máscara se muestra en la figura 11.

Figura 11. Máscara para detección del iris

Fuente: Autor del proyecto

El máximo valor del resultado de la correlación entre esta máscara y la imagen con detección de bordes, dará un punto localizado en el interior del iris, permitiendo la segmentación correcta del ojo, como se muestra en la figura 12.

Figura 12. Detección del iris y segmentación del ojo



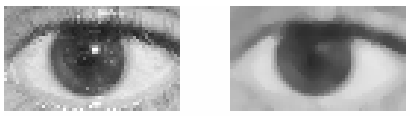
Fuente: Autor del proyecto

2.4. RECONOCIMIENTO DE IMÁGENES

Para reconocer que la imágen segmentada corresponde con la imágen de un ojo abierto, se utiliza el coeficiente de correlación y una imágen patrón. Antes de calcular el coeficiente de correlación se suaviza la imágen con un filtro mediana de 5x5 y se aplica un algoritmo de compensación de iluminación que permita uniformizar los valores de iluminación en la imágen.

En la figura 13 se muestra el resultado de aplicar filtrado de mediana en un entorno 5x5 a la imágen segmentada del ojo.

Figura 13. Filtrado de mediana



Fuente: Autor del proyecto

2.4.1. Algoritmo de Compensación de Iluminación. Para compensación de iluminación se utiliza el algoritmo propuesto por Ilknow Park, Jung-Ho y Hyeran Byun en [3], donde se calcula la intensidad promedio *AI* de la imágen segmentada P_{MN} de tamaño *M x N* píxeles.

$$AI = \frac{1}{M \cdot N} \cdot \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} P_{ij}$$

Para que la intensidad promedio *AI* sea igual a la intensidad promedio esperada *EAI*, esta debe ser multiplicada por un factor *V*. De acuerdo con [3] la intensidad promedio esperada *EAI* se ajusto empíricamente en un valor de 190.

$$EAI = V \cdot AI$$

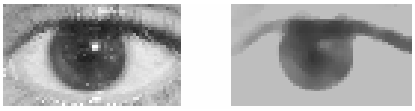
Despejando el factor *V* se tiene:

$$V = \frac{EAI}{AI}$$

Cada píxel de la imágen segmentada P_{MN} es multiplicado por el factor *V* obtenido de la ecuación anterior con el fin de incrementar la intensidad. Si el valor de esta intensidad aumentada es mayor que el valor de *EAI*, este valor se reemplaza por el valor de *EAI*.

Al aplicar el filtro mediana seguido por el algoritmo de compensación de iluminación a la imágen del ojo se obtiene el resultado ilustrado en la figura 14.

Figura 14. Compensación de iluminación



Fuente: Autor del proyecto

2.4.2. Coeficiente de Correlación. El coeficiente de correlación se basa en el mismo concepto de correlación descrito anteriormente, es independiente a los cambios de escala aplicados a la intensidad de *f(x,y)* y *w(x,y)* y está normalizado en el rango -1 a 1. Para imágenes de máxima correspondencia el coeficiente de correlación es 1 y para imágenes de mínima correspondencia -1. Matemáticamente se define como:

$$Y(s,t) = \frac{\sum_x \sum_y [f(x,y) - \overline{f}(x,y)] \cdot [w(x-s,y-t) - \overline{w}(x,y)]}{\left[\sum_x \sum_y [f(x,y) - \overline{f}(x,y)]^2 \sum_x \sum_y [w(x-s,y-t) - \overline{w}(x,y)]^2 \right]^{1/2}}$$

Donde $s = 0,1,2,...,M-1$, $t = 0,1,2,...,N-1$, $\overline{w}$ *(x,y)* es el valor medio de intensidad de w(x,y), $\overline{f}(x,y)$ es el valor medio de $f(x,y)$ en la región coincidente con la actual ubicación de *w* y las sumatorias se calculan para las coordenadas comunes a *f* y *w*.

Para simplificar los cálculos se escoge una imágen patrón del mismo tamaño que la imágen segmentada, de esta forma el coeficiente de correlación será solo un valor.

Para reconocer tanto ojos completamente abiertos, como parcialmente abiertos se emplearon las imágenes patrón mostradas en la figura 15 donde el fondo corresponde a una intensidad de 190 de acuerdo al valor tope del algoritmo de

compensación de iluminación y la intensidad del iris y la pupila se seleccionaron experimentalmente dando valores de 75 y 25 respectivamente.

Figura 15. Imágenes patrón para el reconocimiento de imágenes correspondientes a ojos

a) **b)**

Fuente: Autor del proyecto

Como frontera de decisión para determinar si la imágen corresponde a un ojo abierto se tomo un valor de 0.5 tanto del coeficiente de correlación *Y1* obtenido con la imágen patrón de la figura 15a) como del coeficiente de correlación *Y2* obtenido con la imágen patrón de la figura 15b).

Si *Y1* es mayor que 0.5 ó *Y2* es mayor que 0.5 se considera que la imágen segmentada *Is* corresponde a la de un ojo abierto.

Y1 > 0.5 ó *Y2* > 0.5 $\Leftrightarrow$ *Is* corresponde a un ojo abierto

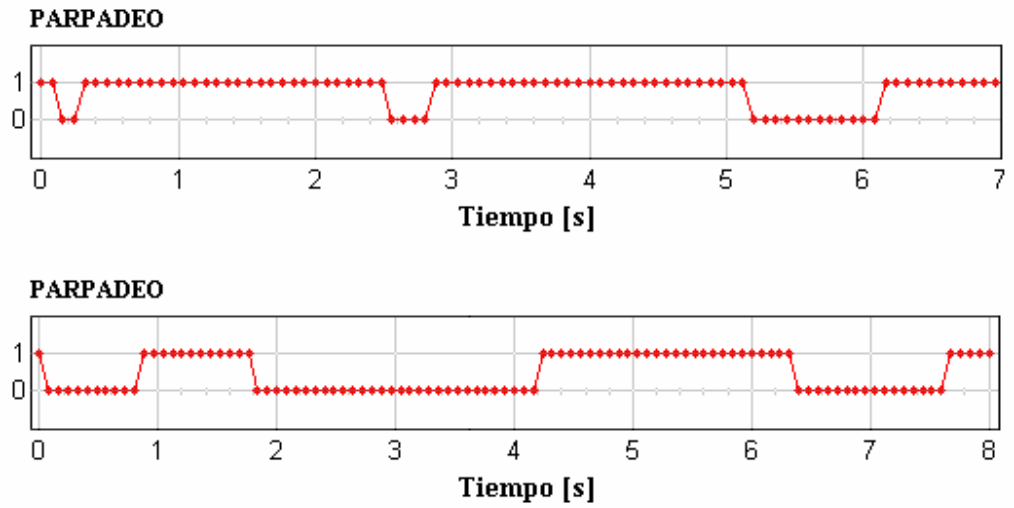
2.5. APLICACIÓN DEL ALGORITMO A SECUENCIAS DE IMÁGENES

Para caracterizar el cansancio y la somnolencia se debe tener en cuenta la duración del parpadeo dentro de un intervalo de tiempo, el cual se puede estimar aplicando el algoritmo en una secuencia de imágenes en la cual el tiempo de muestreo con que se adquieren las imágenes debe ser mayor que el tiempo de ejecución del algoritmo sobre cada imágen. Contabilizando el número de *frames* *Nf* durante los cuales los ojos de la persona permanecen cerrados se puede calcular el tiempo del parpadeo *tp* de acuerdo con la siguiente relación.

$$t_p = t_m \cdot (Nf - 1)$$

En la figura 16 se muestran 2 gráficas obtenidas al procesar secuencias de imágenes de 12.5 fps. Cada punto corresponde a un *frame* ó imágen analizada, donde el valor 1 corresponde a los ojos abiertos y 0 a los ojos cerrados.

Figura 16. Resultado del algoritmo al procesar secuencias de imágenes



Fuente: Autor del proyecto

2.5.1. Perclos. Según estudios realizados en [9], [10], [11], [12] es una medida aceptada como índice psicó-fisiológico de cansancio y somnolencia y como un método de detección de estados de alerta. Perclos (de su nombre en ingles *percent closure*) mide el porcentaje de un periodo de tiempo determinado en que los ojos de una persona permanecen cerrados.

$$\text{PERCLOS} = \frac{tc \cdot 100\%}{tb}$$

Donde *tc* es el tiempo que permanece los ojos cerrados y *tb* es el tiempo base para calcular este porcentaje

En [13] para un tiempo base de 180 segundos se estima como valor umbral de estados de alerta porcentajes mayores a 0.12% lo que equivale a periodos de cierre mayores a 2.15 segundos. En el desarrollo de este proyecto se toma como tiempo base 3 segundos y como valor umbral porcentajes mayores al 70% dando un tiempo de cierre de 2.1 segundos.

3. RESULTADOS EXPERIMENTALES

Para evaluar el algoritmo propuesto se usó una base de datos de 137 imágenes a color en el modelo RGB, con un tamaño de 640x480 píxeles y 256 niveles de intensidad por plano de color. Para la adquisición de las imágenes se usó una cámara digital Sony modelo DSC-P93A. Las imágenes fueron adquiridas dentro de un vehículo con diversas condiciones de iluminación, diferentes distancias entre la cámara y la persona. Personas en distintas posiciones y con la mirada en diferentes direcciones, con los ojos abiertos y con los ojos cerrados. En la toma de imágenes participaron 19 personas de diferentes tonalidades de piel y ojos de diferentes colores. En la figura 17 se muestran algunas de las imágenes de la base de datos con diferentes características.

Figura 17. Imágenes de la base de datos

Fuente: Autor del proyecto

El desarrollo del algoritmo se realizó en MATLAB 7.1 (Matrix Laboratory), el cual como su nombre lo indica es un entorno de cómputo científico cuyo lenguaje está enfocado al manejo matricial. Cuenta con un *toolbox* (librería) para el procesamiento digital de imágenes y con *simulink* una herramienta para la simulación de sistemas. En el anexo A se encuentran los códigos de los algoritmos implementados en este entorno.

Los tiempos de ejecución de los algoritmos, calculados por MATLAB fueron obtenidos en un PC con procesador Dual Core Intel de 3.4 GHz y memoria RAM de 1GB.

3.1. SEGMENTACIÓN DEL ROSTRO

Este algoritmo se aplicó a las imágenes de la base de datos, dando como resultado en todos los 137 casos una subimágen del área de los ojos en el rostro. Caracterizándose por un tiempo de ejecución máximo de 0.0314 segundos, un tiempo de ejecución mínimo de 0.0228 segundos y un tiempo de ejecución promedio de 0.0285 segundos. Estos datos se registran en la tabla 1. En la figura 18 se muestran los resultados de algunas imágenes segmentadas.

Tabla 1. Resultados del algoritmo de segmentación del rostro

SEGMENTACIÓN DEL ROSTRO	
Total Imágenes Analizadas	137
Segmentación Exitosa	137
Segmentación Incorrecta	0
Porcentaje Acierto	100%
Tiempos Ejecución	
Total Imágenes Analizadas	137
Tiempo Ejecución Máximo [s]	0.0314
Tiempo Ejecución Mínimo [s]	0.0228
Tiempo Ejecución Promedio [s]	0.0285

Fuente: Autor del proyecto

Figura 18. Imágenes de rostros segmentados

Fuente: Autor del proyecto

3.2. DETECCIÓN DEL IRIS

Aplicando el algoritmo de segmentación del rostro seguido del algoritmo de detección del iris a las 95 imágenes en las cuales las personas se encuentran con los ojos abiertos, se logró una adecuada detección en 93 de estas imágenes dando un porcentaje de acierto de 97.89%. Calculando los tiempos de ejecución en la base de datos se obtuvo un tiempo de ejecución máximo de 0.0453 segundos, un tiempo de ejecución mínimo de 0.0326 segundos y un tiempo de ejecución promedio de 0.0388 segundos como se consigna en la tabla 2. En la figura 19 se muestran algunos ejemplos de detección del iris y segmentación del ojo.

Tabla 2. Resultados del algoritmo de detección del iris

DETECCIÓN DEL IRIS	
Total Imágenes Analizadas	95
Detección Exitosa	93
Detección Incorrecta	2
Porcentaje Acierto	97.89%
Tiempos Ejecución	
Total Imágenes Analizadas	137
Tiempo Ejecución Máximo [s]	0.0453
Tiempo Ejecución Mínimo [s]	0.0326
Tiempo Ejecución Promedio [s]	0.0388

Fuente: Autor del proyecto

Figura 19. Ejemplos de detección del iris y segmentación del ojo

Fuente: Autor del proyecto

3.3. RECONOCIMIENTO DE IMÁGENES

Para la validación de este algoritmo se calcularon las Ratas de Falso Acierto y Falso Rechazo, la sensibilidad ó fracción de verdaderos aciertos y la especificidad ó fracción de verdaderos rechazos. Se considera un falso acierto cuando el algoritmo indica que los ojos están cerrados, cuando en realidad se encuentran abiertos. Se considera un falso rechazo cuando el algoritmo indica que los ojos

están abiertos, cuando en realidad se encuentran cerrados. Se considera un verdadero acierto cuando el algoritmo indica correctamente que los ojos están cerrados. Se considera un verdadero rechazo cuando el algoritmo indica correctamente que los ojos están abiertos. Sobre la base de datos en estudio, se obtuvieron 3 falsos aciertos, 4 falsos rechazos, 38 verdaderos aciertos y 92 verdaderos rechazos, dando ratas de falso acierto y falso rechazo de 2.19% y 2.92%, respectivamente y valores de sensibilidad y especificidad de 90.48% y 96.84%, respectivamente. Los tiempos de ejecución del algoritmo completo son de 0.0740 segundos para el tiempo máximo, 0.0605 segundos para el tiempo mínimo y 0.0675 segundos para el tiempo promedio. Todos estos datos se registran en la tabla 3.

Tabla 3. Resultados del algoritmo de reconocimiento de imágenes

RECONOCIMIENTO DE IMÁGENES	
Total de Verificaciones (Tv)	137
Verdadero Aciertos (VA)	38
Verdadero Rechazos (VR)	92
Falsos Aciertos (FA)	3
Falsos Rechazos (FR)	4
$FAR = \frac{FA}{Tv} \cdot 100\,\%$	2.19%
$FRR = \frac{FR}{Tv} \cdot 100\,\%$	2.92%
$Sensibilidad = \frac{VA}{VA+FR} \cdot 100\%$	90.48%
$Especificidad = \frac{VR}{VR+FA} \cdot 100\%$	96.84%
Tiempos Ejecucion	
Total Imágenes Analizadas	137
Tiempo Ejecución Máximo [s]	0.0740
Tiempo Ejecución Mínimo [s]	0.0605
Tiempo Ejecución Promedio [s]	0.0675

Fuente: Autor del proyecto

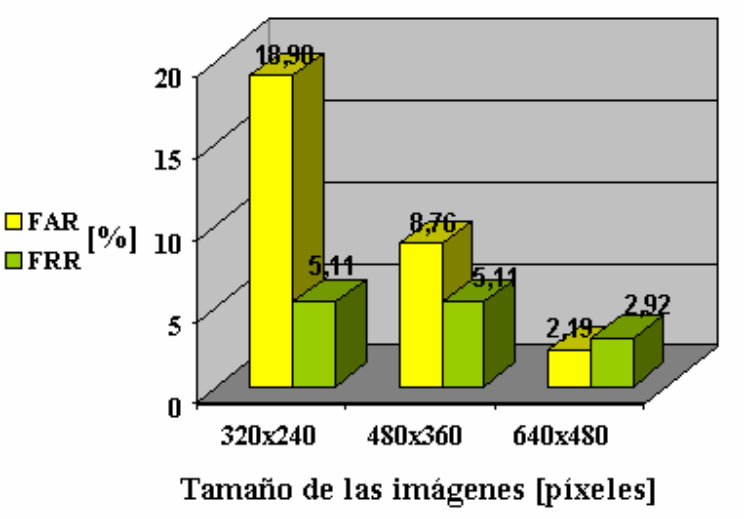
Con el fin de evaluar el rendimiento del algoritmo con imágenes de tamaños estándar y determinar el tamaño optimo de las imágenes de entrada, se modificó el tamaño de las imágenes de la base de datos a 480x360 píxeles y 320x240 píxeles respectivamente. Los resultados obtenidos con estas imágenes justifican la selección de 640x480 píxeles como el tamaño adecuado de las imágenes de entrada. Estos resultados se muestran en la tabla 4. En la figura 20 se grafican los resultados de las ratas de falso acierto y falso rechazo. En la figura 21 se grafican los resultados de la sensibilidad y la especificidad.

Tabla 4. Comparación de resultados de acuerdo al tamaño de las imágenes

COMPARACION DE RESULTADOS DE ACUERDO AL TAMAÑO DE LAS IMÁGENES			
Tamaño Imágenes [píxeles]	640x480	480x360	320x240
Total de Verificaciones (Tv)	137	137	137
Verdadero Aciertos (VA)	38	35	35
Verdadero Rechazos (VR)	92	83	69
Falsos Aciertos (FA)	3	12	26
Falsos Rechazos (FR)	4	7	7
$FAR = \frac{FA}{Tv} \cdot 100\%$	2.19%	8.76%	18.98%
$FRR = \frac{FR}{Tv} \cdot 100\%$	2.92%	5.11%	5.11%
$Sensibilidad = \frac{VA}{VA+FR} \cdot 100\%$	90.47%	83.33%	83.33%
$Especificidad = \frac{VR}{VR+FA} \cdot 100\%$	96.84%	87.37%	72.63%

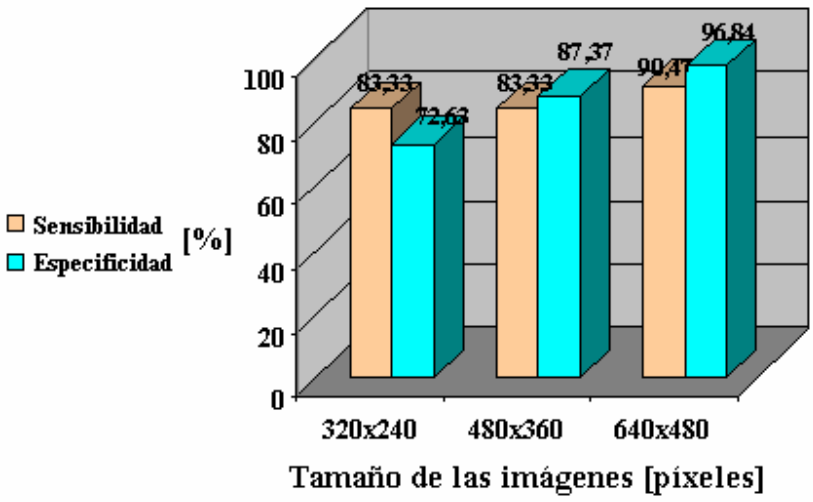
Fuente: Autor del proyecto

Figura 20. Gráfica de las ratas de falso acierto y falso rechazo de acuerdo al tamaño de las imágenes



Fuente: Autor del proyecto

Figura 21. Gráfica de la sensibilidad y la especificidad de acuerdo al tamaño de las imágenes



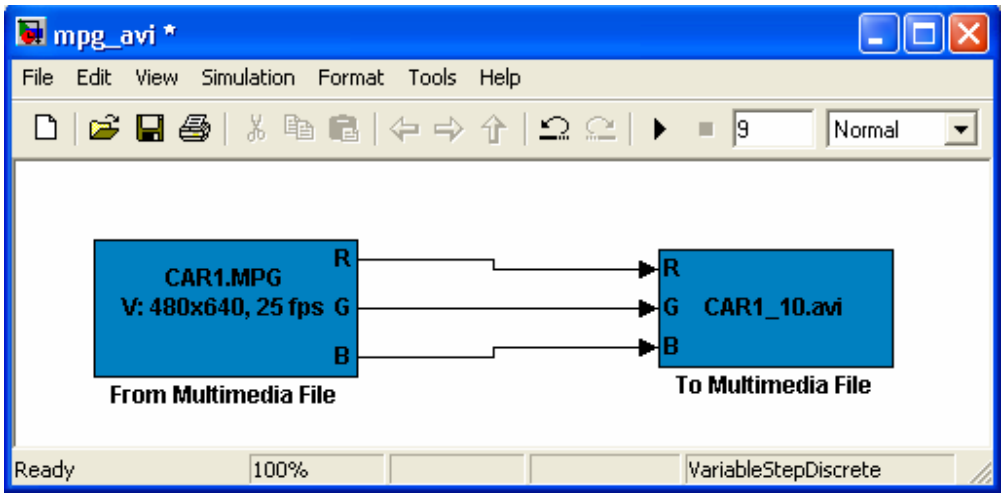
Fuente: Autor del proyecto

3.4. SECUENCIAS DE IMÁGENES

Para evaluar el funcionamiento del algoritmo con secuencias de imágenes se utilizó también una cámara digital Sony modelo DSC-P93A. Se registraron 8 videos con una frecuencia de muestreo de 25 fps en formato *MPG* (Formato estándar de compresión de video), con tiempos de duración entre los 6 y 10 segundos, en los cuales se muestran diferentes personas dentro de un vehículo simulando estados de somnolencia.

Para descomprimir los videos originales y pasarlos a formato *AVI* se emplearon los bloques de la herramienta *Simulink* que se muestran en la siguiente figura:

Figura 22. Bloques para descomprimir y cambiar el formato de video



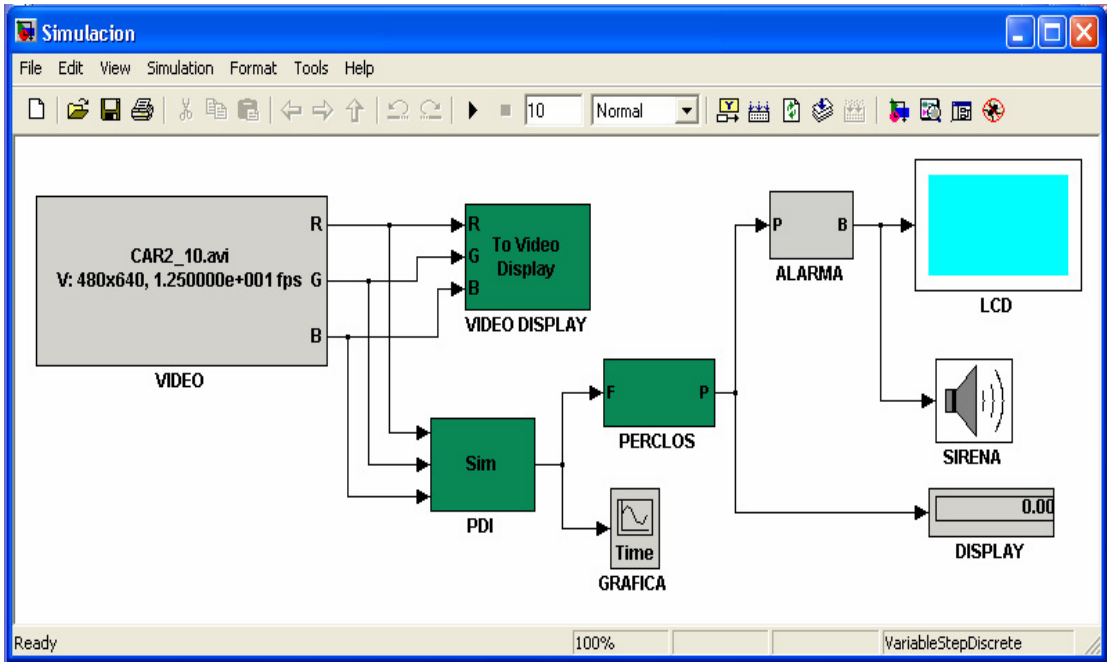
Fuente: Autor del proyecto

Para muestrear los videos en formato *AVI* y obtener secuencias de imágenes de 12.5 fps, que equivale a un tiempo de muestreo de 0.08 segundos, el cual es superior a los tiempos de ejecución mostrados en el numeral anterior, se utilizaron los siguientes comandos:

```
>> F=aviread('CAR1_10.avi',[1:2:251])
>> mov2avi(F,'CAR1_10.AVI','compression','none','fps',12.5)
```

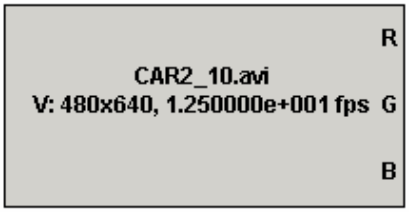
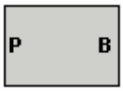
Para simular el funcionamiento del algoritmo con secuencias de imágenes se creó el sistema que se muestra en la siguiente figura:

Figura 23. Sistema para simular el algoritmo con secuencias de imágenes



Fuente: Autor del proyecto

A continuación se explica la función que cumple cada bloque dentro del sistema:

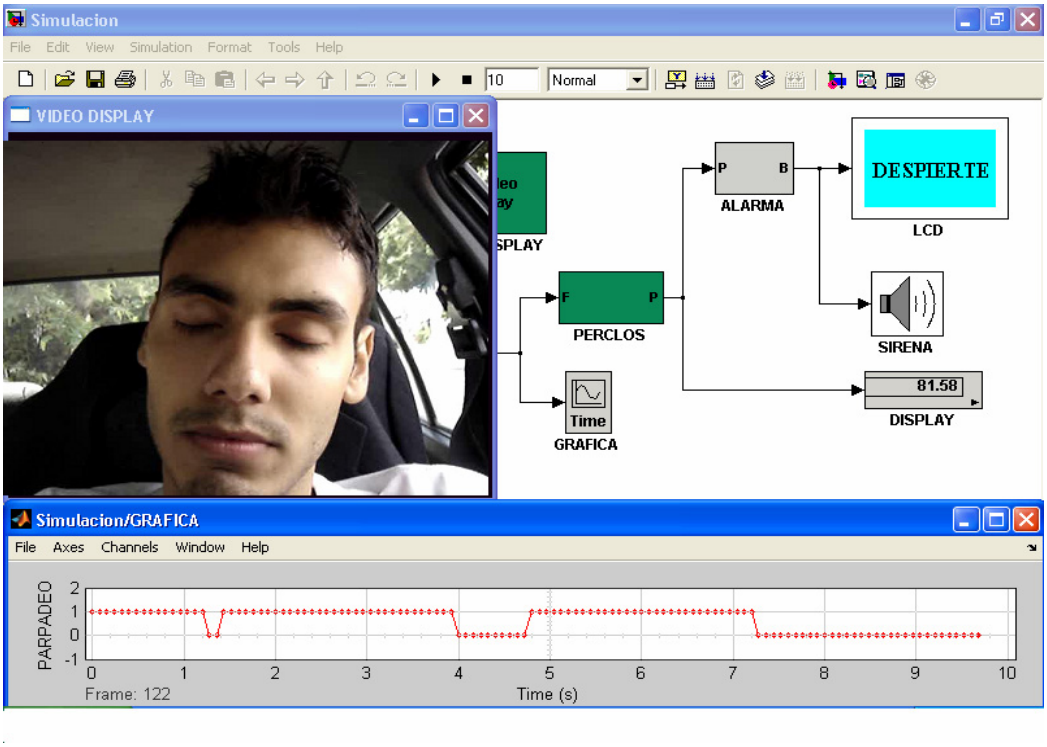
 VIDEO	- Lee la secuencia de imágenes de un archivo en formato multimedia. - En el cuadro de opciones se debe especificar el tiempo de muestreo el cual es de 0.08 segundos.
 VIDEO DISPLAY	- Despliega una ventana para visualizar la secuencia de imágenes seleccionada en el bloque VIDEO.
 PDI	- Este bloque realiza la rutina de procesamiento digital de imágenes para la detección del parpadeo.
 GRAFICA	- Este bloque grafica el resultado de cada frame en un diagrama de tiempo. - En el cuadro de opciones se debe especificar el numero total de frames de la secuencia de imágenes.
 PERCLOS	- Realiza el calculo de PERCLOS cada vez que se obtiene el resultado de un *frame*. - El calculo se realiza con un registro FIFO en el cual se almacenan los resultados de los últimos 38 *frames* los cuales equivalen a un tiempo base de 3 segundos.
 ALARMA	- Activa las alarmas visuales y auditivas cuando el porcentaje de PERCLOS excede el 70%.
 DISPLAY	- Visualiza el porcentaje de PERCLOS calculado en cada *frame*.

 LCD	- Despliega un mensaje de alerta.
 SIRENA	- Produce un sonido de alerta.

Para realizar la simulación en tiempo real se debe ajustar el tiempo de simulación igual al tiempo de duración de la secuencia de imágenes seleccionada, en el cuadro de opciones del bloque VIDEO ajustar el tiempo de muestreo en 0.08 segundos y en el cuadro de opciones del bloque GRAFICA especificar el número total de *frames* de la secuencia de imágenes. Para iniciar la simulación se presiona la tecla PLAY.

En la figura 24 se muestra una simulación en proceso. En la gráfica se visualiza el número de *frames* analizados, cada punto representa un *frame*, 1 indica que los ojos están abiertos y 0 indica que los ojos están cerrados. El bloque DISPLAY muestra el valor de PERCLOS en ese instante para una base de tiempo de 3 segundos y el bloque LCD despliega el mensaje de alerta.

Figura 24. Simulación con secuencias de imágenes



Fuente: Autor del proyecto

De las 8 secuencias de imágenes utilizadas, en 7 de ellas todos los *frames* fueron procesados correctamente y en una secuencia los resultados fueron erróneos debido a fallas en la detección de bordes y en la detección del iris. Estos resultados se muestran en la tabla 5.

Tabla 5. Resultados con secuencias de imágenes

SECUENCIAS DE IMAGENES	
Numero Total Secuencias	8
Secuencias Procesadas Correctamente	7
Secuencias Procesadas Incorrectamente	1

Fuente: Autor del proyecto

4. CONCLUSIONES

Se implementó un algoritmo basado en técnicas de procesamiento digital de imágenes el cual se constituye como un método no intrusivo para detectar el parpadeo de los ojos utilizando secuencias de imágenes biométricas. Este algoritmo consta de tres etapas: segmentación del rostro, detección del iris y reconocimiento de imágenes.

Los resultados de los tiempos de ejecución obtenidos (0.0740 segundos para el tiempo máximo, 0.0605 segundos para el tiempo mínimo y 0.0675 segundos para el tiempo promedio) permite la utilización de estos algoritmos en un sistema en tiempo real con secuencias de imágenes de hasta 12.5 fps.

Los resultados experimentales (FAR=2.19% FRR=2.92% Sensibilidad=90.47% Especificidad=96.84%) demuestran la eficacia del algoritmo, en situaciones con diferentes distancias de ubicación de la cámara, diferentes posiciones de la persona, diferentes tonalidades de piel, diferentes colores de ojos, complejidad del fondo y condiciones de iluminación (excepto casos nocturnos sin ninguna fuente de iluminación).

Una de las situaciones de mayor restricción encontradas en este trabajo es el uso de anteojos por parte del individuo. Durante la segmentación del rostro el marco de los anteojos elimina la conectividad entre la parte superior e inferior del rostro por lo tanto se segmentan partes del rostro en las que no están incluidos los ojo y la pigmentación de los lentes o el reflejo de la luz en estos impide que se puedan detectar con claridad los bordes del iris.

De acuerdo a los tamaños estándar con los que las cámaras digitales adquieren las imágenes, se encontró que el tamaño más adecuado para el algoritmo

implementado es de 640x480 píxeles, ya que tamaños mayores implican mayor coste computacional y mayor tiempo de ejecución. Con imágenes de menor tamaño al efectuar detección de bordes el borde del iris pierde definición lo cual reduce la precisión y confiabilidad del algoritmo.

5. RECOMENDACIONES

Evaluar la alternativa de adquirir las imágenes por medio de una cámara infrarroja o la alternativa de instalar alguna fuente de luz dentro del vehículo que no incida directamente en el rostro del conductor o cause alguna molestia o distracción a este, para contrarrestar condiciones extremas de iluminación (imágenes ó casos nocturnos).

Extender el número de imágenes y secuencias de imágenes de la base de datos y evaluar el uso de sistemas de clasificación y reconocimiento de patrones característicos para las aberturas de los ojos tales como redes neuronales o inteligencia artificial, con el fin de aumentar la robustez del algoritmo.

Estudiar cuidadosamente el sitio de ubicación y la posición de la cámara dentro del vehículo, generando un protocolo de registro para tener en cuenta los posibles movimientos del conductor que impidan el registro adecuado del rostro y los ojos dentro de la imágen.

Realizar un estudio exhaustivo sobre los posibles índices psicológicos y fisiológicos de somnolencia y cansancio y sobre todas las posibles variantes de este complejo comportamiento.

REFERENCIAS BIBLIOGRÁFICAS

[1] D. Chai and K. N. Ngan, “Face Segmentation Using Skin-Color Map in Videophone Aplications”, IEEE transactions on circuits and systems for video technology, vol. 9, no. 4, june 1999

[2] T. D’Orazio, M. Leo, G. Cicirelli, A. Distante, “An Algorithm for real time eye detection in face images”, Institute of Intelligent Systems for Automation. Bari, Italy.

[3] Ilknow Park, Jung-Ho, Hyeran Byun "Efficient Measurement of Eye Blinking Under Various Illumination Conditions for Drowsiness Detection System". Dept. of Computer Science, Yonsei University, Seoul, Korea, 2006.

[4] Gonzáles Rafael C, Woods Richard E, “Tratamiento Digital de Imágenes”, Avisón-Desleí/Díaz de Santos, 1996.

[5] Gonzalez Rafael C, Woods Richard E, ”Digital Image Processing”, Second Edition, Prentice Hall, 2002.

[6] Gonzales Rafael C, Woods Richard E, Eddins Steven L, ”Digital Image Processing using MATLAB”, Pearson Prentice Hall, 2004.

[7] Pajares Gonzalo, de la Cruz Jesus M, “Vision por Computador Imágenes Digitales y Aplicaciones”, Alfaomega RA-MA, 2002.

[8] Oppenheim Alan V, Schafer Ronald W, Buck Jhon R, “Tratamiento de Señales en Tiempo Discreto”, Segunda Edicion, Prentice Hall, 2000.

[9] Federal Highway Administration, Office of motors Carriers U.S, “PERCLOS: A Valid Psychophysiological Measure of Alertness As Assessed by Psychomotor Vigilante” Octubre 1998.

[10] National Highway Traffic Safety Administration, U.S. Department of Transportation, “Evaluation of Techniques for Ocular Measurement as an Index of Fatigue and the Basis for Alertness Management” , Abril 1998.

[11] Takchito Hayami "Detecting Drowsiness while Driving by Measuring Eye Movement- A Pilot Study". The IEEE 5th International Conference on Intelligent Transportation Systems; 3-6 November 2002, Singapore.

[12] Takahiro Hamada "Detecting Methods for Drivers' Drowsiness Applicable to Individual Features". IEEE paper, 2004.

[13] National Highway Traffic Safety Administration, U.S. Department of Transportation, “A Preliminary Assessment of Algorhitms for Drowsy and Inattentive Driver Detection on the Road” Marzo 1999.

ANEXOS

ANEXO A. CÓDIGOS DE LOS ALGORITMOS IMPLEMENTADOS EN MATLAB

En este anexo están todos los códigos de las m-function de los algoritmos implementados en el desarrollo del proyecto. Las funciones se encuentran organizadas alfabéticamente.

C

```
% CALCULO DEL COEFICIENTE DE CORRELACION
% SUBRUTINA
% O ES LA IMAGEN SEGMENTADA
% B,E SON CONSTANTES CALCULADAS CON LA IMAGEN PATRON
% Y ES EL COEFICIENTE DE CORRELACION

function Y=C_corr(O,B,E)

O=single(O);
Om=mean2(O);

A=O-Om;
C=conv2(A,B,'valid');
D=sum(sum(A.^2));

if D==0
    Y=-1;
else
    Y=C/(D*E)^0.5;
end
```

```
%ALGORITMO COMPENSACION ILUMINACION
%SUBRUTINA
%Is ES LA IMAGEN DE ENTRADA
%Ic ES LA IMAGEN CON COMPENSACION DE ILUMINACION

function Ic=c_ilum(Is)

EAI=190;
AI=1+mean2(Is);
T=EAI/AI;
Ic=Is*T;
Ic=imadjust(Ic,[0 190/255],[0 190/255]);
```

```
% ALGORITMO PARA MODIFICAR EL TAMAÑO DE LAS IMAGENES
% LAS IMAGENES ORIGINALES SE DESIGNAN CON LAS LETRAS E,F
% LAS NUEVAS IMAGENES SE DESIGNAN CON LAS LETRAS G,H

function Camb_Tam

for k=1:42

    I='E_';
    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    Ir=imread(I);
    Ir=imresize(Ir,3/4);
    I(1)='G';
    imwrite(Ir,I)
end

for k=1:95

    I='F_';
    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    Ir=imread(I);
    Ir=imresize(Ir,3/4);
    I(1)='H';
    imwrite(Ir,I)
end



% METODO DE CRECIMIENTO DE REGIONES POR AGREGACION DE PIXELES
% SUBRUTINA
% Ib ES UNA IMAGEN BINARIA DE ENTRADA
% P ES EL PUNTO GENERADOR

function B=cre_reg(Ib,P)

L=bwlabeln(Ib,4);
```

```
B=ismember(L,L(P(1),P(2)));
```

D

```
%ALGORITMO DETECCION IRIS PARA IMAGENES 640x480
%TIEMPOS DE EJECUCION Y VISUALIZACION RESULTADOS
%I ES EL NOMBRE DE LA IMAGEN SE DEBE INGRESAR ENTRE COMILLAS
%EJ: DET_IRIS('C_1.JPG')

function DET_IRIS(I)

[R Is Y1 X1]=SEG_ROS_S(I);
P1=[-1 -1 -1;0 0 0;1 1 1];
P2=[-1 0 1;-1 0 1;-1 0 1];
S1=[-1 -2 -1;0 0 0;1 2 1];
S2=[-1 0 1;-2 0 2;-1 0 1];
W=[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3 -1 -1 -1 -1 -1  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1;
   -1 -1 -1 -1  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3 -1 -1 -1 -1;
   -1 -1 -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -1 -1 -1;
   -1 -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -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 -1  3  3  3  3 -1 -1;
   -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -1;
   -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -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 -1 -1 -1  3  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;];
Ws=single(W);

%PREWIT
tic
D1=imfilter(Is,P1);
D2=imfilter(Is,P2);
G=D1+D2;
G1=single(im2bw(G,0.4));
C1=conv2(G1,Ws,'valid');
M1=max(max(C1));
[Y1 X1]=find(C1==M1);
t1=toc

%SOBEL
tic
D1=imfilter(Is,S1);
D2=imfilter(Is,S2);
G=D1+D2;
G2=single(im2bw(G,0.5));
C2=conv2(G2,Ws,'valid');
M2=max(max(C2));
[Y2 X2]=find(C2==M2);
t2=toc
```

```
figure
imshow(Is)
hold on
plot(X2+15,Y2+8,'r.','linewidth',9)
```

%ALGORITMO DETECCION IRIS PARA IMAGENES DE 640x480
%SUBRUTINA

```
function [Xc Yc]=DET_IRIS_S(Is,X1,Y1)

S1=[-1 -2 -1;0 0 0;1 2 1];
S2=[-1 0 1;-2 0 2;-1 0 1];
W=[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1  3  3  3  3  3  3  3 -1 -1 -1 -1 -1  3  3  3  3  3  3  3 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3  3 -1 -1 -1 -1 -1;
   -1 -1 -1 -1  3  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3  3 -1 -1 -1 -1;
   -1 -1 -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -1 -1 -1;
   -1 -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -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 -1  3  3  3  3 -1 -1;
   -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1 -1;
   -1 -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -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 -1 -1 -1  3  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;
   -1  3  3  3 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  3  3  3 -1;];
Ws=single(W);

%SOBEL
D1=imfilter(Is,S1);
D2=imfilter(Is,S2);
G1=D1+D2;
G2=single(im2bw(G1,0.5));
C=conv2(G2,Ws,'valid');
M=max(max(C));
[Y X]=find(C==M);
Xc=X(1)+X1+15;
Yc=Y(1)+Y1+8;
```

%ALGORITMO DETECCION IRIS PARA IMAGENES DE 480x360
%SUBRUTINA

```
function [Xc Yc]=DET_IRIS_S2(Is,X1,Y1)

S1=[-1 -2 -1;0 0 0;1 2 1];
S2=[-1 0 1;-2 0 2;-1 0 1];
W=[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1  2  2  2  2  2  2  2  2  2  2  2 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2 -1 -1 -1 -1 -1;
   -1 -1 -1 -1  2  2  2  2  2  2 -1 -1 -1 -1 -1  2  2  2  2  2  2 -1 -1 -1 -1;
```

```
   -1 -1 -1  2  2  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2  2  2 -1 -1 -1;
   -1 -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1 -1 -1;
   -1 -1  2  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2  2 -1 -1;
   -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1 -1;
   -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2 -1 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2 -1];
Ws=single(W);

%SOBEL
D1=imfilter(Is,S1);
D2=imfilter(Is,S2);
G1=D1+D2;
G2=single(im2bw(G1,0.5));
C=conv2(G2,Ws,'valid');
M=max(max(C));
[Y X]=find(C==M);
Xc=X(1)+X1+12;
Yc=Y(1)+Y1+7;
```

%ALGORITMO DETECCION IRIS PARA IMAGENES DE 320x240
%SUBRUTINA

```
function [Xc Yc]=DET_IRIS_S3(Is,X1,Y1)

S1=[-1 -2 -1;0 0 0;1 2 1];
S2=[-1 0 1;-2 0 2;-1 0 1];
W=[-1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1 -1 -1  2  2  2  2  2 -1 -1 -1 -1 -1 -1 -1;
   -1 -1 -1 -1 -1 -1  2  2  2  2  2  2  2  2  2 -1 -1 -1 -1 -1;
   -1 -1 -1 -1  2  2  2  2  2  2  2  2  2  2  2  2 -1 -1 -1 -1;
   -1 -1 -1  2  2  2  2  2 -1 -1 -1 -1 -1  2  2  2  2  2 -1 -1;
   -1 -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1 -1;
   -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1;
   -1 -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1;
   -1  2  2  2 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1 -1  2  2  2 -1];
Ws=single(W);

%SOBEL
D1=imfilter(Is,S1);
D2=imfilter(Is,S2);
G1=D1+D2;
G2=single(im2bw(G1,0.5));
C=conv2(G2,Ws,'valid');
M=max(max(C));
[Y X]=find(C==M);
Xc=X(1)+X1+10;
Yc=Y(1)+Y1+6;
```

F

```
% ALGORITMO PARA CALCULAR FAR EN IMAGENES DE 640x480

function FAR

I='F_';
FA=0;

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    D=MACRO_S(I);

    if D==0
        FA=FA+1;
        V{FA,:}=I;
    end
end

V
FA
FAR=FA/137*100
```

```
% ALGORITMO PARA CALCULAR FAR EN IMAGENES DE 480x360

function FAR_2

I='H_';
FA=0;

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';
```

```
    D=MACRO_S2(I);

    if D==0
        FA=FA+1;
        V{FA,:}=I;
    end
end

V
FA
FAR=FA/137*100
```

% ALGORITMO PARA CALCULAR FAR EN IMAGEES DE 320x240

```
function FAR_3

I='D_';
FA=0;

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    D=MACRO_S3(I);

    if D==0
        FA=FA+1;
        V{FA,:}=I;
    end
end

V
FA
FAR=FA/137*100
```

% ALGORITMO PARA CALCULAR FFR EN IMAGENES DE 640X480

```
function FRR
```

```
I='E_';
FR=0;

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    D=MACRO_S(I);

    if D==1
        FR=FR+1;
        V{FR,:}=I;
    end
end

V
FR
FRR=FR/137*100
```

% ALGORITMO PARA CALCULAR FFR EN IMAGENES DE 480X360

```
function FRR_2

I='G_';
FR=0;

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    D=MACRO_S2(I);

    if D==1
        FR=FR+1;
        V{FR,:}=I;
```

```
    end
end

V
FR
FRR=FR/137*100
```

% ALGORITMO PARA CALCULAR FFR EN IMAGENES DE 320x240

```
function FRR_3

I='C_';
FR=0;

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    D=MACRO_S3(I);

    if D==1
        FR=FR+1;
        V{FR,:}=I;
    end
end

V
FR
FRR=FR/137*100
```

M

% ALGORITMO COMPLETO PARA IMAGENES DE 640X480
% TIEMPO DE EJECUCION Y VISUALIZACION RESULTADOS
% I ES EL NOMBRE DE LA IMAGEN SE DEBE INGRESAR ENTRE COMILLAS
% EJ: MACRO('C_1.JPG')

```
function MACRO(I)

%CONSTANTES
Op1=single(imread('OJO_3.BMP'));
```

```
Op2=single(imread('M_OJO.BMP'));

Opm1=mean2(Op1);
Opm2=mean2(Op2);

B1=Op1-Opm1;
B2=Op2-Opm2;

E1=sum(sum(B1.^2));
E2=sum(sum(B2.^2));

%PROCESO
tic

[R Is Y1 X1]=SEG_ROS_S(I);
[Xc Yc]=DET_IRIS_S(Is,X1,Y1);
O=R(Yc-25:Yc+10,Xc-30:Xc+30);
Om=medfilt2(O,[5 5],'symmetric');
Om=c_ilum_2(Om);
Y1=C_corr(Om,B1,E1);
Y2=C_corr(Om,B2,E2);

D1=Y1>0.5;
D2=Y2>0.5;
D=D1|D2;

t=toc

%VISUALIZACION

figure
imshow(R)
hold on
plot(Xc,Yc,'r.','linewidth',5)
if D==1
    title('OJOS ABIERTOS')
else
    title('OJOS CERRADOS')
end

figure
imshow(Is)

figure
imshow(O)
```

```
% ALGORITMO COMPLETO PARA IMAGENES DE 640X480
% SUBRUTINA

function [D t]=MACRO_S(I)

%CONSTANTES
Op1=single(imread('OJO_3.BMP'));
Op2=single(imread('M_OJO.BMP'));

Opm1=mean2(Op1);
Opm2=mean2(Op2);

B1=Op1-Opm1;
B2=Op2-Opm2;

E1=sum(sum(B1.^2));
E2=sum(sum(B2.^2));

%PROCESO

tic

[R Is Y1 X1]=SEG_ROS_S(I);
[Xc Yc]=DET_IRIS_S(Is,X1,Y1);
O=R(Yc-25:Yc+10,Xc-30:Xc+30);
O=medfilt2(O,[5 5],'symmetric');
O=c_ilum_2(O);
Y1=C_corr(O,B1,E1);
Y2=C_corr(O,B2,E2);

D1=Y1>0.5;
D2=Y2>0.5;
D=D1|D2;

t=toc;
```

```
% ALGORITMO COMPLETO PARA IMAGENES DE 480X360
% SUBRUTINA

function D=MACRO_S2(I)

%CONSTANTES
Op1=single(imread('OJO2_3.BMP'));
Op2=single(imread('M2_OJO.BMP'));

Opm1=mean2(Op1);
Opm2=mean2(Op2);

B1=Op1-Opm1;
```

```
B2=Op2-Opm2;

E1=sum(sum(B1.^2));
E2=sum(sum(B2.^2));

%PROCESO

[R Is Y1 X1]=SEG_ROS_S2(I);
[Xc Yc]=DET_IRIS_S2(Is,X1,Y1);
O=R(Yc-19:Yc+7,Xc-22:Xc+23);
O=medfilt2(O,[5 5],'symmetric');
O=c_ilum(O);
Y1=C_corr(O,B1,E1);
Y2=C_corr(O,B2,E2);

D1=Y1>0.5;
D2=Y2>0.5;
D=D1|D2;
```

% ALGORITMO COMPLETO PARA IMAGENES DE 320X240
% SUBRUTINA

```
function D=MACRO_S3(I)

%CONSTANTES
Op1=single(imread('OJO3_3.BMP'));
Op2=single(imread('M3_OJO.BMP'));

Opm1=mean2(Op1);
Opm2=mean2(Op2);

B1=Op1-Opm1;
B2=Op2-Opm2;

E1=sum(sum(B1.^2));
E2=sum(sum(B2.^2));

%PROCESO

[R Is Y1 X1]=SEG_ROS_S3(I);
[Xc Yc]=DET_IRIS_S3(Is,X1,Y1);
O=R(Yc-12:Yc+5,Xc-15:Xc+15);
O=medfilt2(O,[3 3],'symmetric');
O=c_ilum(O);
Y1=C_corr(O,B1,E1);
Y2=C_corr(O,B2,E2);

D1=Y1>0.5;
D2=Y2>0.5;
D=D1|D2;
```

S

```
%ALGORITMO SEGMENTAR ROSTRO BASADO COLOR PIEL
%VISUALIZACION DE RESULTADOS
%I ES EL NOMBRE DE LA IMAGEN SE DEBE INGRESAR ENTRE COMILLAS
%EJ: SEG_ROS('C_1.JPG')

function SEG_ROS(I)

% CONSTANTES
X=[64;
   64;
   48;
   80];

Y=[32;
   48;
   48;
   48];

%PROCESO
tic
Ir=imread(I);
Ic=imresize(Ir,0.2);

R=single(Ic(:,:,1))/255;
G=single(Ic(:,:,2))/255;
B=single(Ic(:,:,3))/255;
Cb=-37.797*R-74.203*G+112*B+128;
Cr=112*R-93.786*G-18.214*B+128;

C=[Cb(32,64);
   Cb(48,64);
   Cb(48,48);
   Cb(48,80)];

[V Ix]=min(C);
P(1)=Y(Ix);
P(2)=X(Ix);

Tcb=abs(Cb-Cb(P(1),P(2)));
B1=Tcb<=9;

Tcr=abs(Cr-Cr(P(1),P(2)));
B2=Tcr<=9;

B=B1&B2;
Ib=cre_reg(B,P);

[y x]=find(Ib);
Y1=5*min(y)+20;
Y2=5*max(y);
```

```
Y2=round(0.6*(Y1+Y2));

D=round(0.6*(Y2-Y1));
X1A=5*min(x);
X2A=5*max(x);
Xo=round(0.5*(X1A+X2A));
X1B=Xo-D;
X2B=Xo+D;
X1=max(X1A,X1B);
X2=min(X2A,X2B);

R=Ir(:,:,1);
Is=R(Y1:Y2,X1:X2);

t=toc

% VISUALIZACION

figure
imshow(Ic)
hold on
plot(X,Y,'r.','linewidth',3)

figure
imshow(Ib)
hold on
plot(P(2),P(1),'r.','linewidth',3)

figure
imshow(Is)
```

%ALGORITMO SEGMENTAR ROSTRO PARA IMAGENES DE 640x480
%SUBRUTINA

```
function [R Is Y1 X1]=SEG_ROS_S(I)

X=[64;
   64;
   48;
   80];

Y=[32;
   48;
   48;
   48];

Ir=imread(I);
Ic=imresize(Ir,0.2);
```

```
R=single(Ic(:,:,1))/255;
G=single(Ic(:,:,2))/255;
B=single(Ic(:,:,3))/255;
Cb=-37.797*R-74.203*G+112*B+128;
Cr=112*R-93.786*G-18.214*B+128;

C=[Cb(32,64);
   Cb(48,64);
   Cb(48,48);
   Cb(48,80)];

[V Ix]=min(C);
P(1)=Y(Ix);
P(2)=X(Ix);

Tcb=abs(Cb-Cb(P(1),P(2)));
B1=Tcb<=9;

Tcr=abs(Cr-Cr(P(1),P(2)));
B2=Tcr<=9;

B=B1&B2;
Ib=cre_reg(B,P);

[y x]=find(Ib);
Y1=5*min(y)+20;
Y2=5*max(y);
Y2=round(0.6*(Y1+Y2));

D=round(0.5*(Y2-Y1));
X1A=5*min(x);
X2A=5*max(x);
Xo=round(0.5*(X1A+X2A));
X1B=Xo-D;
X2B=Xo+D;
X1=max(X1A,X1B);
X2=min(X2A,X2B);

R=Ir(:,:,1);
Is=R(Y1:Y2,X1:X2);
```

%ALGORITMO SEGMENTAR ROSTRO PARA IMAGENES DE 480x360
%SUBRUTINA

```
function [R Is Y1 X1]=SEG_ROS_S2(I)

X=[64;
   64;
   48;
   80];
```

```
Y=[32;
   48;
   48;
   48];

Ir=imread(I);
Ic=imresize(Ir,4/15);

R=single(Ic(:,:,1))/255;
G=single(Ic(:,:,2))/255;
B=single(Ic(:,:,3))/255;
Cb=-37.797*R-74.203*G+112*B+128;
Cr=112*R-93.786*G-18.214*B+128;

C=[Cb(32,64);
   Cb(48,64);
   Cb(48,48);
   Cb(48,80)];

[V Ix]=min(C);
P(1)=Y(Ix);
P(2)=X(Ix);

Tcb=abs(Cb-Cb(P(1),P(2)));
B1=Tcb<=9;

Tcr=abs(Cr-Cr(P(1),P(2)));
B2=Tcr<=9;

B=B1&B2;
Ib=cre_reg(B,P);

[y x]=find(Ib);
Y1=round(15/4*min(y))+15;
Y2=round(15/4*max(y));
Y2=round(0.6*(Y1+Y2));

D=round(0.5*(Y2-Y1));
X1A=round(15/4*min(x));
X2A=round(15/4*max(x));
Xo=round(0.5*(X1A+X2A));
X1B=Xo-D;
X2B=Xo+D;
X1=max(X1A,X1B);
X2=min(X2A,X2B);

R=Ir(:,:,1);
Is=R(Y1:Y2,X1:X2);
```

```
%ALGORITMO SEGMENTAR ROSTRO PARA IMAGENES DE 320x240
%SUBRUTINA

function [R Is Y1 X1]=SEG_ROS_S3(I)

X=[64;
   64;
   48;
   80];

Y=[32;
   48;
   48;
   48];

Ir=imread(I);
Ic=imresize(Ir,0.4);

R=single(Ic(:,:,1))/255;
G=single(Ic(:,:,2))/255;
B=single(Ic(:,:,3))/255;
Cb=-37.797*R-74.203*G+112*B+128;
Cr=112*R-93.786*G-18.214*B+128;

C=[Cb(32,64);
   Cb(48,64);
   Cb(48,48);
   Cb(48,80)];

[V Ix]=min(C);
P(1)=Y(Ix);
P(2)=X(Ix);

Tcb=abs(Cb-Cb(P(1),P(2)));
B1=Tcb<=9;

Tcr=abs(Cr-Cr(P(1),P(2)));
B2=Tcr<=9;

B=B1&B2;
Ib=cre_reg(B,P);

[y x]=find(Ib);
Y1=round(2.5*min(y))+10;
Y2=round(2.5*max(y));
Y2=round(0.6*(Y1+Y2));

D=round(0.5*(Y2-Y1));
X1A=round(2.5*min(x));
X2A=round(2.5*max(x));
Xo=round(0.5*(X1A+X2A));
X1B=Xo-D;
```

```
X2B=Xo+D;
X1=max(X1A,X1B);
X2=min(X2A,X2B);

R=Ir(:,:,1);
Is=R(Y1:Y2,X1:X2);
```

```
%ALGORITMO SEGMENTAR ROSTRO
%SUBRUTINA PARA SIMULACION CON SECUENCIAS DE IMAGENES

function [Is Y1 X1]=SEG_ROS_Sim(R,G,B)

X=[64;
   64;
   48;
   80];

Y=[32;
   48;
   48;
   48];

Rr=imresize(R,0.2);
Gr=imresize(G,0.2);
Br=imresize(B,0.2);

Rr=single(Rr)/255;
Gr=single(Gr)/255;
Br=single(Br)/255;
Cb=-37.797*Rr-74.203*Gr+112*Br+128;
Cr=112*Rr-93.786*Gr-18.214*Br+128;

C=[Cb(32,64);
   Cb(48,64);
   Cb(48,48);
   Cb(48,80)];

[V Ix]=min(C);
P(1)=Y(Ix);
P(2)=X(Ix);

Tcb=abs(Cb-Cb(P(1),P(2)));
B1=Tcb<=9;

Tcr=abs(Cr-Cr(P(1),P(2)));
B2=Tcr<=9;

B=B1&B2;
Ib=cre_reg(B,P);
```

```
[y x]=find(Ib);
Y1=5*min(y)+20;
Y2=5*max(y);
Y2=round(0.6*(Y1+Y2));

D=round(0.5*(Y2-Y1));
X1A=5*min(x);
X2A=5*max(x);
Xo=round(0.5*(X1A+X2A));
X1B=Xo-D;
X2B=Xo+D;
X1=max(X1A,X1B);
X2=min(X2A,X2B);

Is=R(Y1:Y2,X1:X2);
```

```
% PROGRAMACION DEL BLOQUE PDI PARA SIMULACION
% ALGORITMO COMPLETO PARA LA DETECCION DEL PARPADEO
% UTILIZANDO SECUENCIAS DE IMAGENES

function Simulacion(block)

setup(block);

function setup(block)

block.NumInputPorts  = 3;
block.NumOutputPorts = 1;

block.SetPreCompInpPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

block.InputPort(1).DatatypeID   = 3;
block.InputPort(1).Complexity   = 'Real';
block.InputPort(1).SamplingMode = 'Sample';
block.InputPort(1).Overwritable = false;
block.InputPort(1).Dimensions   = [480 640];

block.InputPort(2).DatatypeID   = 3;
block.InputPort(2).Complexity   = 'Real';
block.InputPort(2).SamplingMode = 'Sample';
block.InputPort(2).Overwritable = false;
block.InputPort(2).Dimensions   = [480 640];

block.InputPort(3).DatatypeID   = 3;
block.InputPort(3).Complexity   = 'Real';
block.InputPort(3).SamplingMode = 'Sample';
block.InputPort(3).Overwritable = false;
block.InputPort(3).Dimensions   = [480 640];
```

```
block.OutputPort(1).DatatypeID   = 1;
block.OutputPort(1).Complexity   = 'Real';
block.OutputPort(1).SamplingMode = 'Sample';
block.OutputPort(1).Dimensions   = 1;

block.NumDialogPrms     = 0;

block.SetAccelRunOnTLC(false);

block.RegBlockMethod('Outputs', @Output);


function D=algoritmo(R,G,B)

Op1=single(imread('OJO_3.BMP'));
Op2=single(imread('M_OJO.BMP'));
Opm1=mean2(Op1);
Opm2=mean2(Op2);
B1=Op1-Opm1;
B2=Op2-Opm2;
E1=sum(sum(B1.^2));
E2=sum(sum(B2.^2));


[Is Y1 X1]=SEG_ROS_Sim(R,G,B);
[Xc Yc]=DET_IRIS_S(Is,X1,Y1);
O=R(Yc-25:Yc+10,Xc-30:Xc+30);
O=medfilt2(O,[5 5],'symmetric');
O=c_ilum_2(O);
Y1=C_corr_2(O,B1,E1);
Y2=C_corr_2(O,B2,E2);

D1=Y1>0.5;
D2=Y2>0.5;
D=D1|D2;
D=single(D);

function Output(block)

block.OutputPort(1).Data =
algoritmo(block.InputPort(1).Data,block.InputPort(2).Data,block.InputPort
(3).Data);
```

T

```
% ALGORITMO PARA CALCULAR LOS TIEMPOS DE EJECUCION
% ALGORITMOS SEGMENTACION DEL ROSTRO + DETECCION IRIS
% PARA IMAGENES DE 640x480
```

```
function Te_DI

I='E_';

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    tic
    [R Is Y1 X1]=SEG_ROS_S(I);
    [Xc Yc]=DET_IRIS_S(Is,X1,Y1);
    t=toc;
    T1(k)=t;
end

I='F_';

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    tic
    [R Is Y1 X1]=SEG_ROS_S(I);
    [Xc Yc]=DET_IRIS_S(Is,X1,Y1);
    t=toc;
    T2(k)=t;
end

tmax=max([max(T1) max(T2)])
tmin=min([min(T1) min(T2)])
tprom=(sum(T1)+sum(T2))/137
```

```
% ALGORITMO PARA CALCULAR LOS TIEMPOS DE EJECUCION
% PARA EL ALGORITMO COMPLETO
% PARA IMAGENES DE 640x480

function Te_M

I='E_';

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    [D t]=MACRO_S(I);
    T1(k)=t;
end

I='F_';

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    [D t]=MACRO_S(I);
    T2(k)=t;
end

tmax=max([max(T1) max(T2)])
tmin=min([min(T1) min(T2)])
tprom=(sum(T1)+sum(T2))/137



% ALGORITMO PARA CALCULAR LOS TIEMPOS DE EJECUCION
% PARA EL ALGORITMO DE SEGMENTACION DEL ROSTRO
% PARA IMAGENES DE 640x480

function Te_SR
```

```
I='E_';

for k=1:42

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    tic
    [R Is Y1 X1]=SEG_ROS_S(I);
    t=toc;
    T1(k)=t;
end

I='F_';

for k=1:95

    P=int2str(k);
    Z=length(P);
    I(3:2+Z)=P;

    I(Z+3)='.';
    I(Z+4)='J';
    I(Z+5)='P';
    I(Z+6)='G';

    tic
    [R Is Y1 X1]=SEG_ROS_S(I);
    t=toc;
    T2(k)=t;
end

tmax=max([max(T1) max(T2)])
tmin=min([min(T1) min(T2)])
tprom=(sum(T1)+sum(T2))/137
```