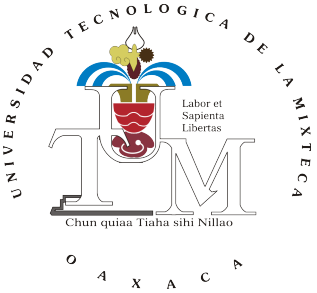
UNIVERSIDAD TECNOLÓGICA DE LA MIXTECA
DIVISIÓN DE ESTUDIOS DE POSGRADO
MAESTRÍA EN ROBÓTICA



Aplicación de técnicas para detección y extracción de iris

Proyecto que se presenta como parte de la materia:

Visión por computadora

Presenta:

Belén Aidee Santiago Marcial

Huajuapan de León, Oaxaca, México, 28 de Junio de 2018

Índice

Índice de figuras

1. Introducción

El proyecto consiste en desarrollar un sistema de identificación biométrica a través del iris de una persona utilizando una PC para realizar el procesamiento de las imágenes. El desarrollo y prueba de los algoritmos se realizó en MATLAB. Las imágenes se extraen digitalmente de una base de datos de internet llamada CASIA (CASIA Iris Image Database v1.0), para extraer la región del iris usan algoritmos de pre-procesamiento de imágenes para poder mejorar la calidad de la imagen y poder extraer de forma eficiente la dicha región.

Las pruebas de los algoritmos para encontrar el radio de la pupila y del iris, se realizaron con una muestra extraída de la base de datos. La base de datos esta compuesta de 749 imágenes, 7 imágenes de 108 individuos distintos.

El documento esta organizado de la siguiente forma: en la sección de marco teórico, se muestran los conceptos que dan sustento a las etapas desarrolladas así como la descripción de cada una de ellas, en la sección de objetivos se muestran el objetivo general y los objetivos específicos que persigue el proyecto desarrollado, en la seccion de desarrollo se describe cada uno de los algoritmos implementados así como los resultados de cada uno de estos, además; también se incluyen pruebas de la implementación del sistema, en las conclusiones se muestran las obcervaciones y resultados obtenidos, por último se incluye la lista de referencias utilizadas en el desarrollo del sistema. Como anexo se muestran los códigos programados para cada una de las etapas.

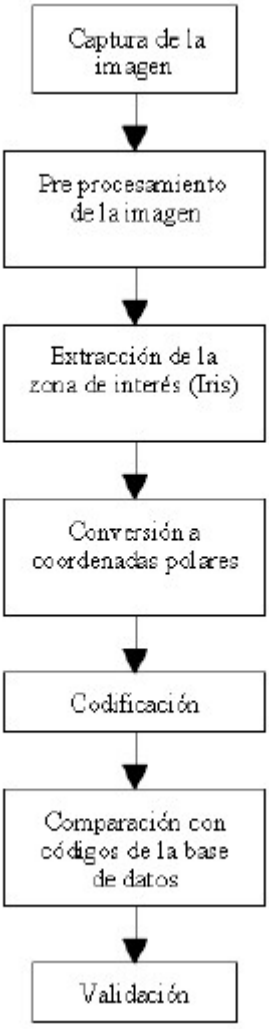


Figura 1: Principio de funcionamiento del reconocimiento de iris.

2. Marco teórico

2.1. Reconocimiento de iris

El reconocimiento del iris utiliza la tecnología de las cámaras: con una fina iluminación infrarroja se reduce el reflejo que se haya podido producir en la convexa córnea y poder crear detalladas imágenes de las complejas estructuras del iris. Una vez convertidas en plantillas digitales, estas imágenes proporcionan una representación matemática del iris, las cuales coinciden con una identificación positiva e inequívoca de un individuo [1].

En este trabajo solo se llegará a la cuarta etapa que es la de conversión a coordenadas polares..

2.1.1. Captura de la imagen

Se captura una imagen de la capa arbórea del iris en blanco y negro, en un entorno correctamente iluminado, usando una cámara de alta resolución. En este caso se tomarán imágenes de la base de datos de CASIA como fue mencionado en la introducción.

2.1.2. Pre-procesamiento de la imagen

A fin de poder extraer la zona de interés (iris) de la imagen capturada, se aplican ciertos filtros sobre la misma. Se aplican algoritmos para aíslar el iris del resto de la imagen. Si el iris no es aislado correctamente puede dificultarse la correcta identificación. Además tambien se normaliza la imagen del iris aislada sea independiente del tamaño de la pupila, debido a que puede variar en cada imagen. Normalizar la imagen implica transformar la región anular del iris en una región rectangular de dimensiones constantes La etapa de mejora prepara a la imagen normalizada modificándola y preparándola para la etapa de extracción [2].

El primer paso es aplicar un filtro de mediana, el cual uniforma la conjuntiva del ojo. Esto sirve para que los bordes de las pestañas, pupila e iris sean más marcados, de tal forma que puedan ser mejor identificados en las siguientes etapas [1].

Hallando el centro y radio de la pupila A la imagen con filtro se le aplica un hestiramiento de histograma para encontrar el ubral y así encontrar las regiones más oscura de la imagen. Se determina el centro y radio de la pupila. Habiendo identificado el círculo de la pupila y sus coordenadas de centro; así como el radio de la misma; la pupila es aislada mediante una máscara de extracción, en la cual el círculo de la pupila es de color negro en un fondo blanco.

Hallando el centro y radio del iris Para extraer el borde externo del iris se aplica estiramiento de histograma de la imagen en escala de grises original, luego se aplican el filtro de mediana para uniformizar las regiones y eliminar falsos contornos, seguidamente se aplican algun operador de detección de bordes, en este caso se aplicará el operador de sobel.

2.1.3. Extracción de la zona de interés

Conociendo las coordenadas del centro del iris y su radio, se traza una circunferencia de color blanco sobre un fondo negro. Se realiza una operación AND entre la imagen captura y las dos mascaras de extracción, consiguiendo aislar la zona de interés (el anillo del iris) en la imagen.

2.1.4. Transformación de coordenadas

Luego de obtener el centro y radio de la pupila y del iris (debido a que se los puede considerar concéntricos [1], aunque no necesariamente lo sean), la región anular obtenida es normalizada a una

imagen rectangular, es decir que la imagen es “estirada”, recorriéndola en sentido antihorario para representarla en el eje polar, realizando las transformaciones correspondientes, para ello se utilizan las operaciones mostradas en [3]. Y por último de hace una tranformación inversa que propone [3], porque la imagen con transformación polar quedan algunos píeles sin información, y al realizar la transformación inversa, el cual, en lugar de recorrer todos los píxeles de la imagen original, recorre todos los píxeles de imagen con transformación polar y busca a que píxeles corresponden de la imágen original.

3. Objetivos

3.1. General

Desarrollar un sistema de deteción de iris y pupila para la extracción de la banda del iris.

3.2. Específicos

- Comprender las diferentes técnicas y algoritmos vistos en clases para el procesamiento digital de imágenes.
- Aplicar operaciones de mejoramiento de imagen para obtener mejores resultados.
- Encontrar el centro de la pupila y del iris.
- Encontrar el radio de la pupila y el radio del iris.
- Extraer la banda de la region del iris.
- Realizar pruebas para validar el algoritmo de extracción de iris.

4. Desarrollo

Se realizó un algoritmo principal (llamado principal.m), en cual se incluyen cada una de las etapas que se describen a continuación.

4.1. Detección y extracción de la región de la pupila

La imagen original se pasa del espacio RGB de 3 dimenciones a escala de gris de dos dimenciones, y queda como se muestra en la figura 2.

Al aplicar el filtro de la mediana a la imagen 2, queda como se muestra en la imagen 3

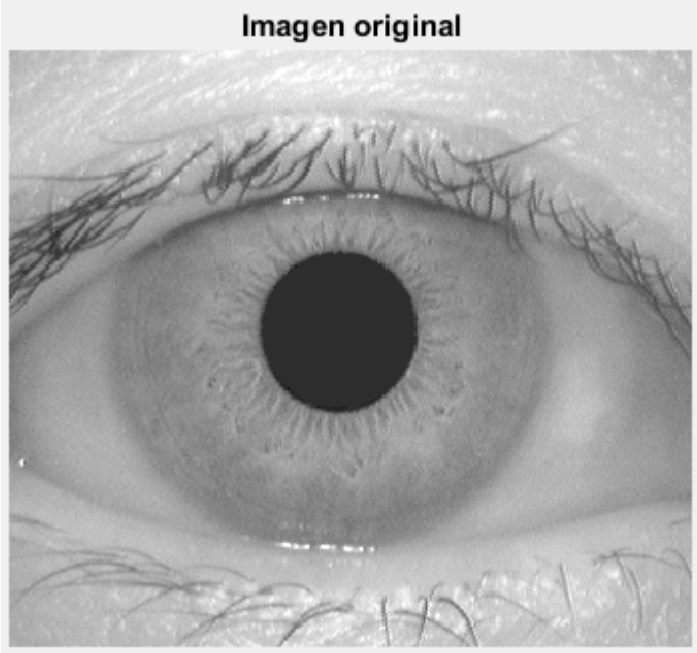


Figura 2: Imagen original de la persona 1.

Figura 3: Imagen con filtro de la mediana de la persona 1.

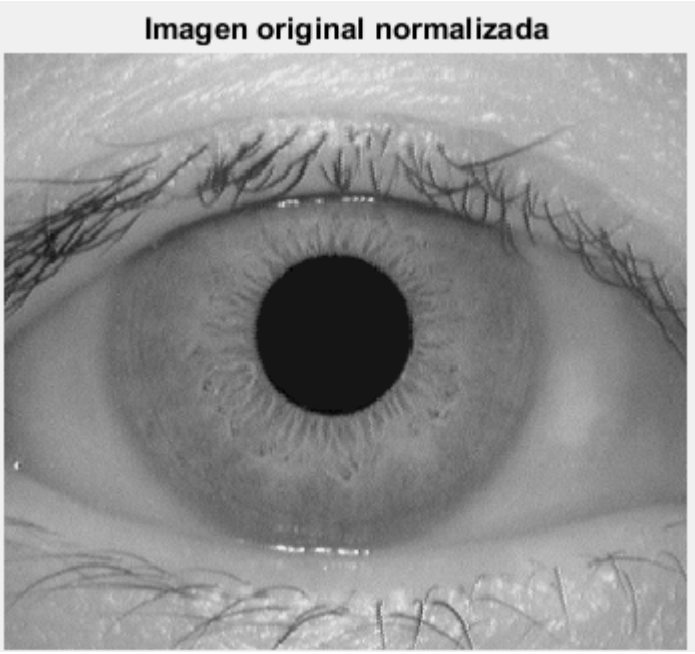


Figura 4: Imagen normalizada de la persona 1.

La imagen filtrada 3 al ser normalizada con el algoritmo normalizar.m de la sección de anexos, queda como se muestra en la figura 4

Se obtienen los histogramas de la imagen de la figura 2, de la imagen de la figura 3, los cuales se muestran en las figuras 4 y 5, respectivamente.

La imagen original de la figura 2 es normalizada para obetener su histograma mostrado en la figura 7, y obcervar, la diferencia entre cada uno de los hisogramas.

La imagen de la figura 3 es normalizada (figura 8) utilizando el algoritmo de normalizar.m de la sección de anexos, para así abarcar toda la región entre 0 y 255, como se muestra en la figura 9.

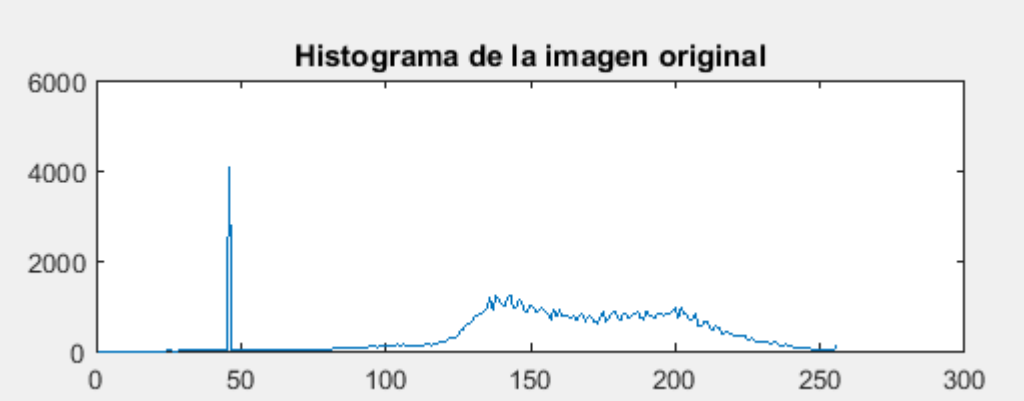


Figura 5: Histograma de la imagen original de la persona 1.

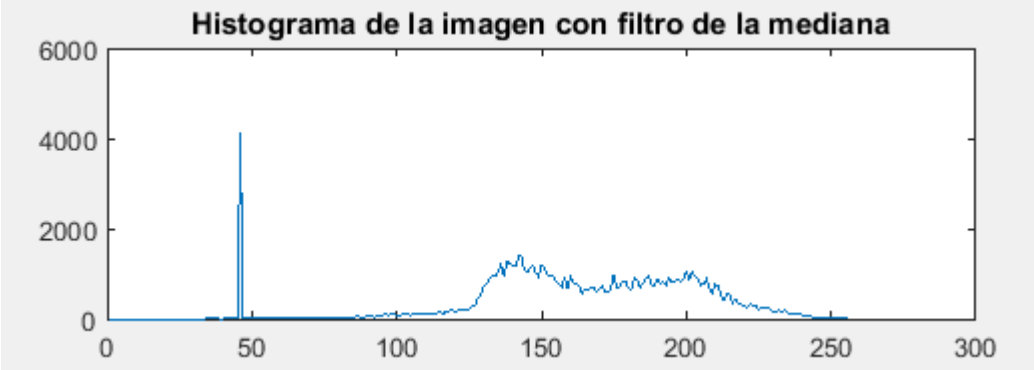


Figura 6: Histograma de la imagen con filtro de la mediana de la persona 1.

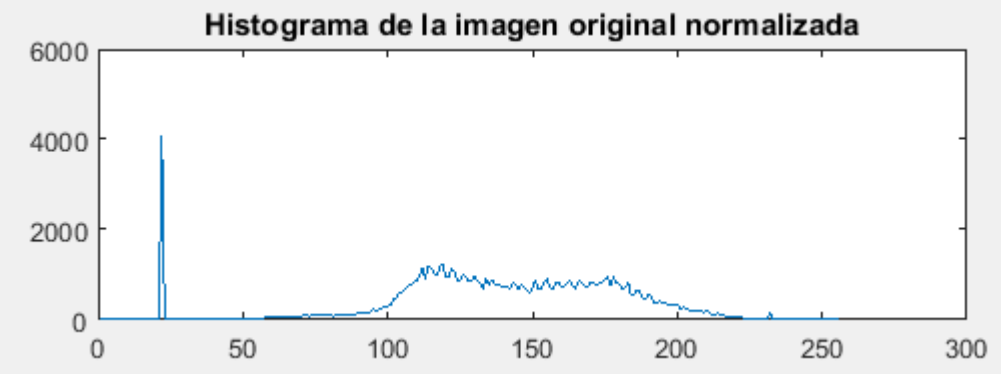


Figura 7: Histograma de la imagen original normalizada de la persona 1.

Figura 8: Imagen con filtro normalizada de la persona 1.

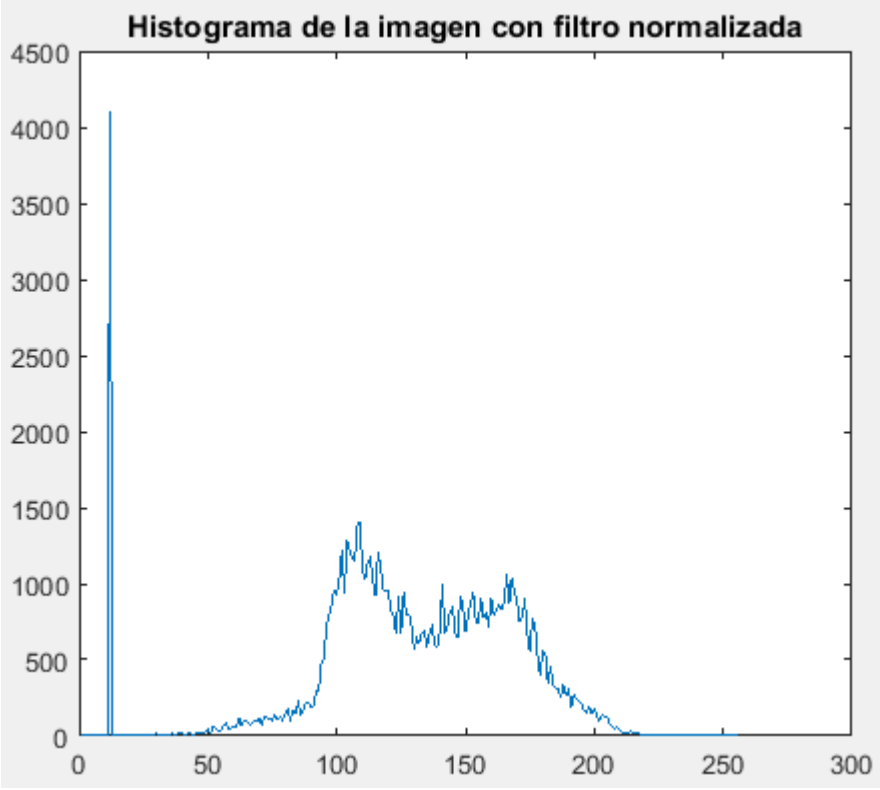


Figura 9: Histograma de la imagen con filtro normalizada de la persona 1.

Figura 10: Imagen filtrada con el valor T Umbral de la persona 1.

Con el histograma de la figura 9, se aplica el algoritmo para detección automática de umbral (umbral.m, mostrado en la sección de anexos), y así restar ese valor a la imagen original para obtener los objetos de la imagen con dicho umbral, como en la figura 10.

A continuación se hace un intercabio de colores de los piexeles, los unos se hacen ceros y los ceros se hacen unos, quedando como se muestra en la figura 11.

Lo que sigue es realizar un etiquetado de los objetos que quedaron después de la unbralización, utilizando el algoritmo etiquetado.m de la sección de anexos. Para ello primero se realiza una binarización de la imagen 11, y luego se realiza el etiquetado de los objetos, figuras 12 y 13, respectivamente.

Con los ojetos etiquetados, se aplica un algoritmo de detección del objeto con el área máxima (archivo area.m, de la sección de anexos), ya que este es la pupila. Una vez encontrado dicho objeto, se realiza un algorimo de deteción de pupila (find_pupila.m de la sección de anexos), que consiste el filtrar la imagen 13 para eliminar el resto de los objetos con menor área como se muestra en la figura 14. Posteriormente, utilizando la función *regionprops* de MATLAB, se obtiene el centro de la pupila en (xc,yc), como se indica en la figura 14.

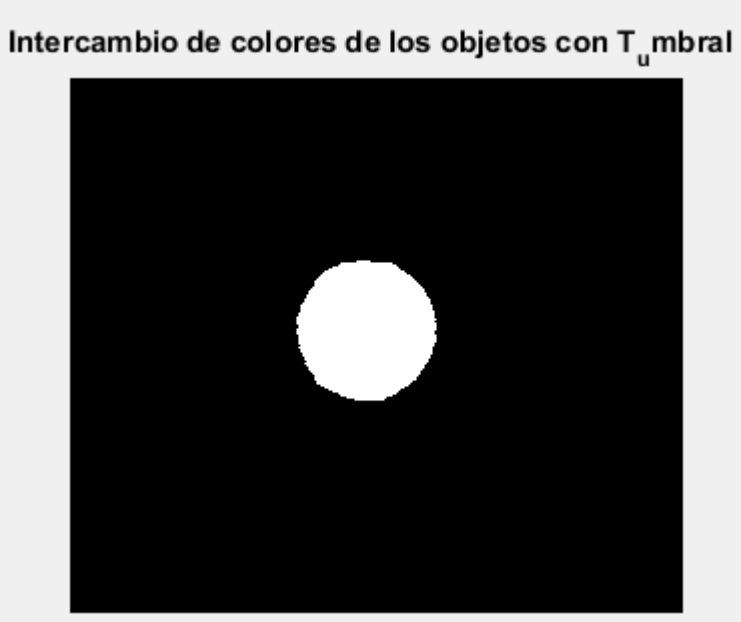


Figura 11: Intercambio de colores de los objetos con T Umbral de la parsona 1.

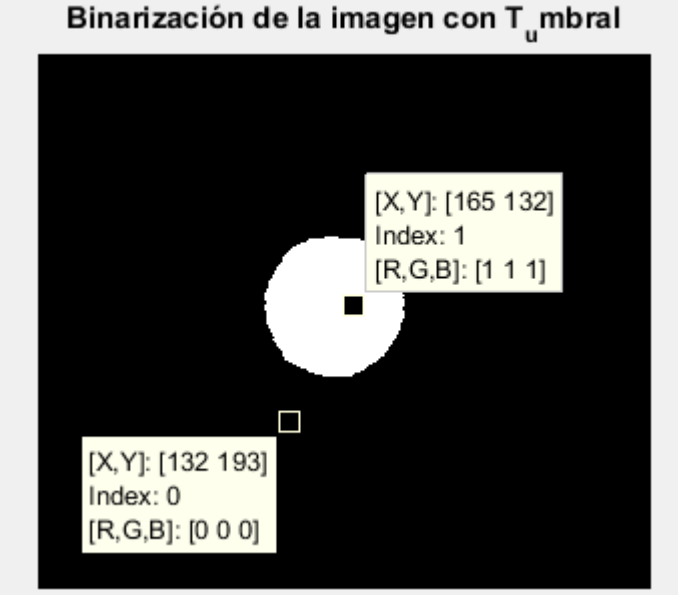


Figura 12: Binarización de la imagen con T Umbral de la persona 1.

Figura 13: Etiquetado de la imagen con T Umbral de la persona 1.

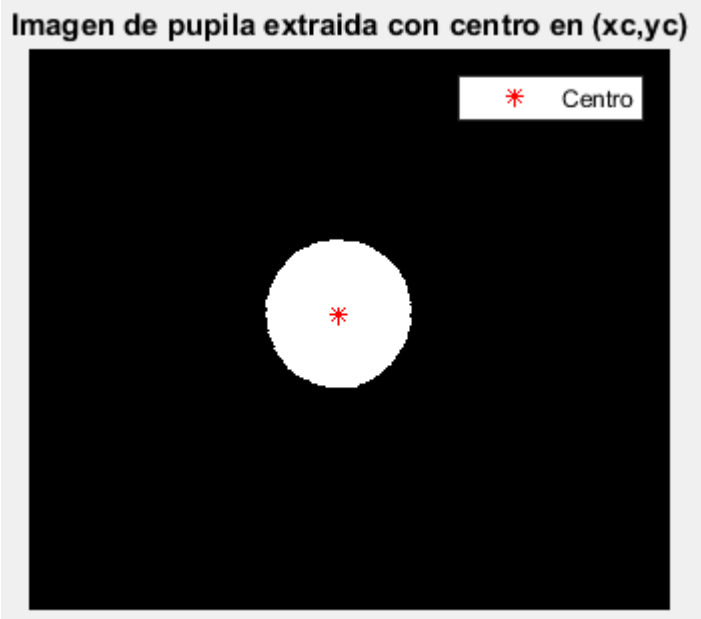


Figura 14: Imagen de la pupila con centro en (Xc,Yc) de la persona1.

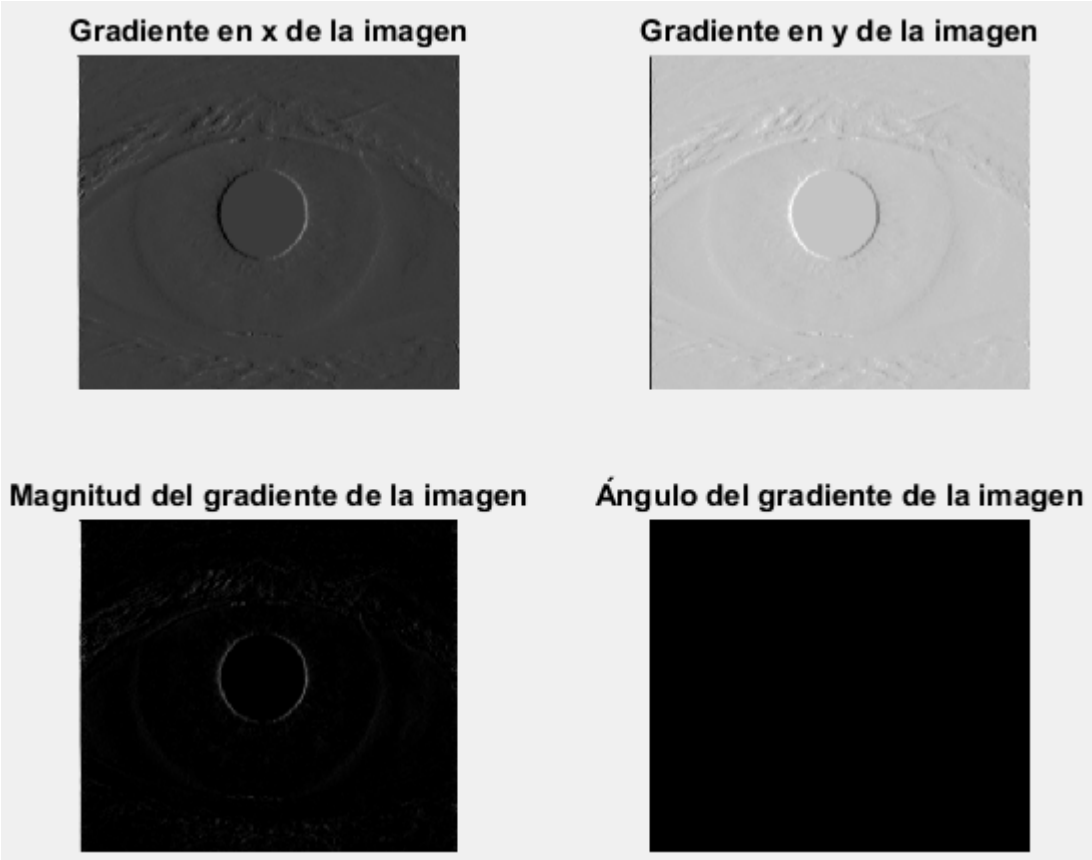


Figura 15: Aplicación del aperador de sobel para extracción de bordes del ojo de la persona 1.

Por último se aplica el algoritmo para encontrar el radio (find_radio.m, de la sección de anexos) de la pupila de la figura 14 y el algoritmo para dibujar un circulo (draw_circle.m, de la sección de anexos), para así encontrar los límites de la pupila.

4.2. Detección y extracción de la región del iris

Primero se aplica un operador de sobel (find_contornos.m de la sección de anexos) para encontrar los bordes de la imagen original 2, como se muestra en la figura 15, una vez encontrada la matriz con la magnitud del gradiente del operador, se aplica un flitrado a dicha matris con un umbral de 15 para encontrar los bordes como se muestra en la figura 5.

Además de la imagen original se obtiene el ruido aplicando el algoritmo find_ruido.m, el cual consiste en obtener el promedio del histograma de la imagen original. Con el valor promedio obtenido,

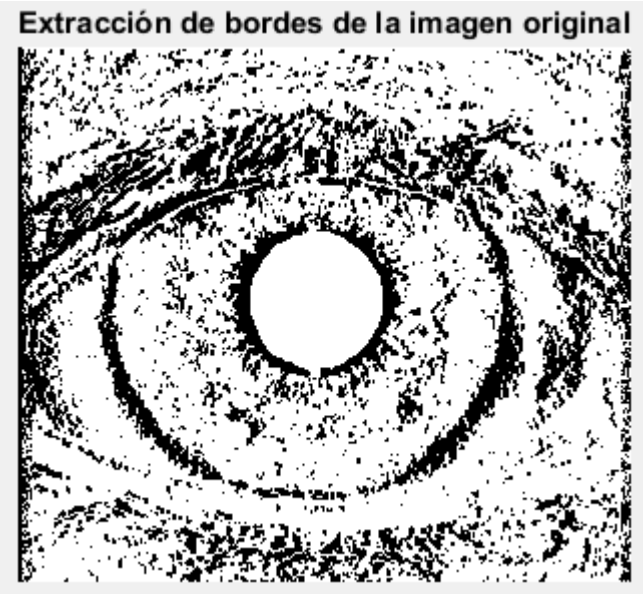


Figura 16: Extracción de bordes de la imagen original de la persona 1.

se realiza un filtro para obtener una nueva imagen con los pixeles con de ese valor, como se muestra en la figura 17.

Enseguida la imagen 17 se resta de la imagen 16 (usando el algoritmo de eliminar_ruido.m de la sección de anexos), para así obtener una imagen con menos ruido, como se muestra en la figura 18.

Posteriormente se aplica una operación morfológica de Top-Hat de tipo disco con doble apertura (algoritmo), una de se=1 y otra de se=2, utilizando el algoritmo *close_open.m*. En la figuras 19 y 20, se muestra la imagen18 con operación de Top-Hat y con doble apertura, respectivamente.

Acontinuación, se hace un recorte rectangular de la figura 20, para eliminar la mayoria de los objetos no importantes. Se utiliza el algoritmo ojo_rectangulo.m de la sección de anexos, el cual consiste en eliminar primero todos los objetos fuera del rectángulo con centro en la pupila, de ancho de dos veces radio de pupila+5 y largo 240. En segundo lugar, eliminar todos los objetos dentro de un rectángulo con centro en la pupila, de largo de dos veces radio de pupila+30 y de ancho de dos veces el radio de la pupila. La imagen resultante queda como se muestra en la figura 21.

Figura 17: Extracción del ruido de la imagen original de la persona 1.

Figura 18: Imagen de los bordes sin ruido de la persona 1.

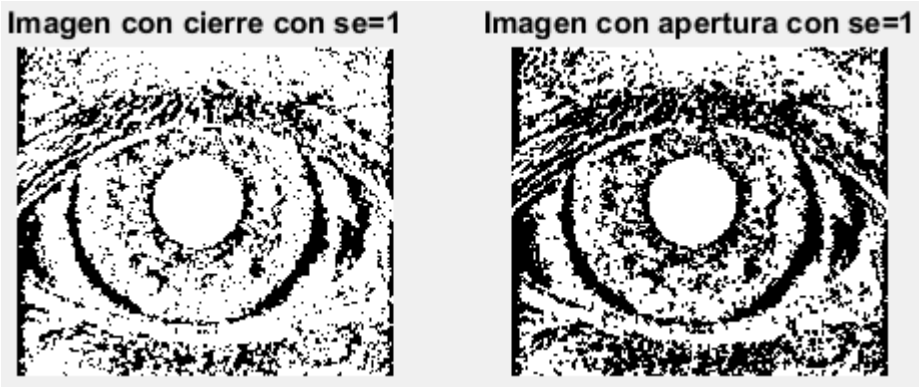


Figura 19: Imagenes con cierre y apertura con se=1 de la persona 1.

Figura 20: Imagen con apertura con se=2 de la persona 1.

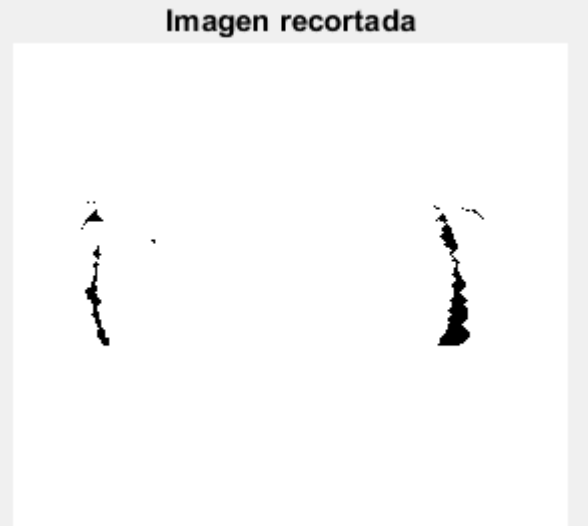


Figura 21: Imagen recortada de la persona 1.

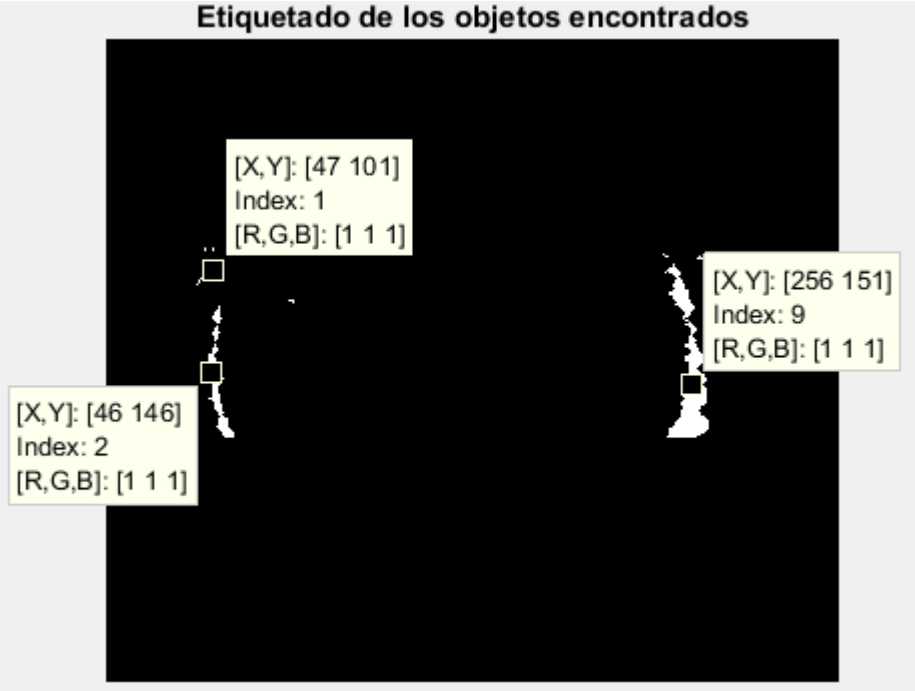


Figura 22: Etiquetado de los objetos encontrados de la persona 1.

El siguiente paso es etiquetar los objetos de la figura 21, utilizando nuevamente el algoritmo etiquetado.m de la sección de anexos, como se muestra en la figura 22.

Con los ojetos etiquetados, se aplica nuevamente el algoritmo de deteccion de área (area.m), pero ahora a la figura 22. Al aplicarlo devuelve un vector con las áreas (vec_I) y una variable (p_1) con la etiqueta del área mayor, esta variable es usada para hacer cero el valor del área de esa etiqueta en el vector. Del vector actualizado se obtiene la posición del segundo objeto con mayor área en la variable p_2. Estas dos etiquetas son utilizadas para filtrar la imagen 22 usando el algoritmo find_iris.m, el cual elimina los objetos que no tengas la etiqueta p_1 o p_2. Así la imagen con los bordes del iris, es la mostrada en la figura 23.

Ahora a la figura 23 se le aplica el algoritmo fin_radio.m, para determinar el tamaño del radio del iris, pero como este algoritmo encuentra el radio de un circulo que encierre todos los objetos de la imagen, se intoduce una vaiable $bandaC = 10$ que se le resta al radio del iris encontrado, para que no sea más grande de lo necesario, que adiferencia de aplicarselo a la pupila, $bandaC = 0$. Una vez obtenido el radio del iris, se aplica el algoritmo de draw_circle, para encontrar el círculo con

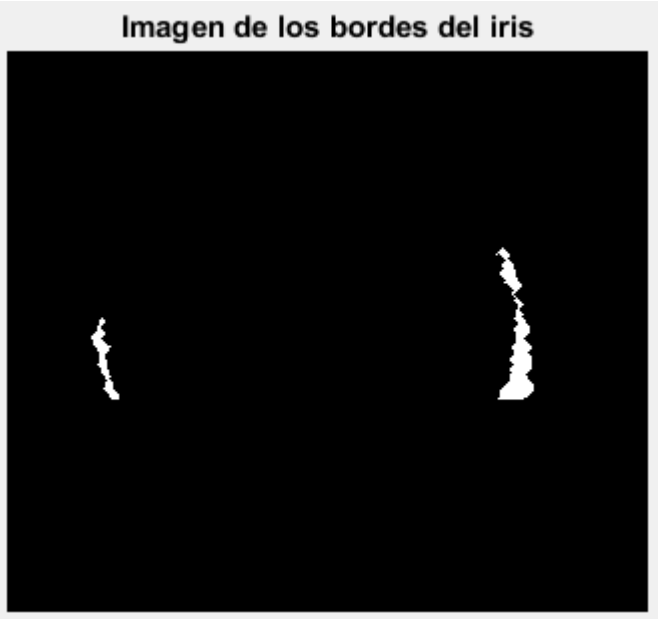


Figura 23: Imagen de los bordes de iris de la persona 1.

centro en (xc,ycE), donde $ycE = yc + 3$, ya que según [1], se puede asumir que tanto el centro de la pupila como el centro del iris, son los mismos, pero [3], dice lo contrario, es por ello que se realizarón varias pruebas y se econtró que la mejor coordena en y para el centro del iris, estaba desfadado de la coordenada en y del centro de la pupila por una distancia de 3. De esta forma se obtuvo la figura 24, en la cual se muestra tanto los límites de pupila, como los del iris, además del centro de la pupila (amarillo) y el centro del iris (verde).

4.3. Extracción de la región del iris

Ahora se aplica el algoritmo de extracción de la región del iris a la imagen original de la figura 2 (extraccion_banda.m), el cual devuelve una matriz con los pixeles entre el radio del iris y de la pupila, como se muestra en la figura 25.

Ensequida se aplica una ecualización por histograma, para resaltar características y suavizar ruido, usando para ello el algoritmo ecualizar.m de la sección de anexos. Y se obtiene la figura 26.

A la figura 5 se le aplica el algoritmo de extracción de la región del iris y esta nueva imagen es ecualizada, quedando como se muestra en la figura 27

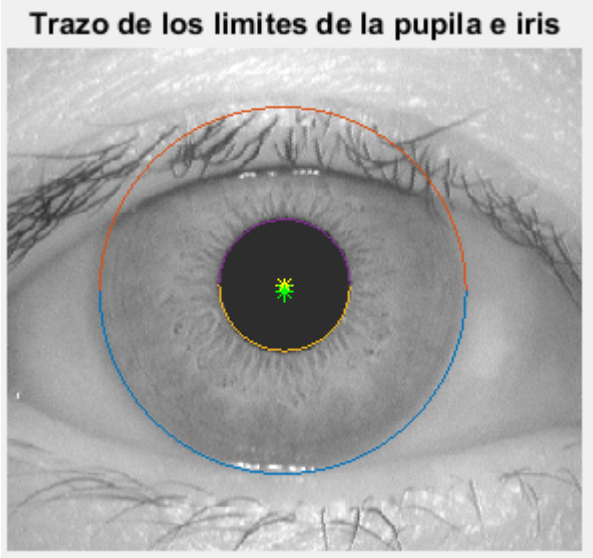


Figura 24: Trazo de los limites de la pupila e iris de la persona 1.

Figura 25: Región del iris de la imagen original de la persona 1.

Figura 26: Región del iris ecualizada de la persona 1.

Figura 27: Región del iris normalizada y ecualizada de la persona 1.

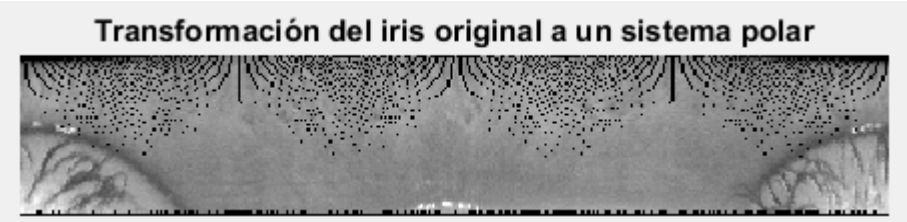


Figura 28: Transformación del iris normalizado a un sistema polar de la persona 1.

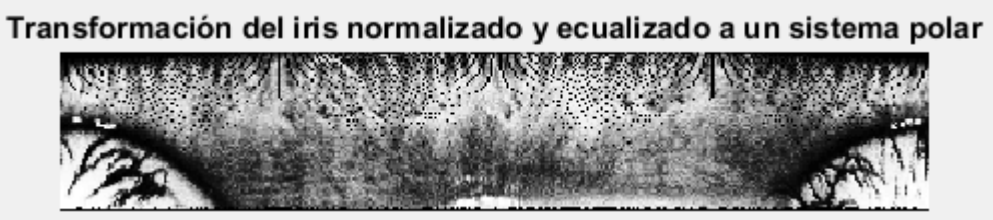


Figura 29: Trasnformación del iris normalizado y ecualizado a un sistema polar de la persona 1.

4.4. Transformación a coordenadas polares

Se desarrolló un algoritmo para transformar la región extrída a coordenadas polares, pues de esta formas normaliza la imagen, para que tengan el mismo tramaño en el eje X, y resulte más facíl, la extracción y comparación de las características. Este algoritmo es trans_PolRec.m que se muestra en la sección de anexos, esta basado en el mostrado en [3] y se le aplica a la figura 25 y 27, las figuras obtenidas son las mostradas en 28 y 29, respectivamente.

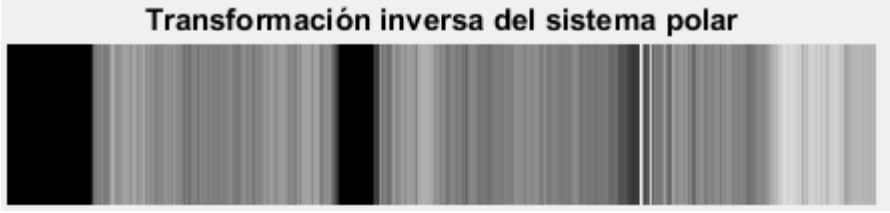


Figura 30: Transformación inversa del sistema polar de la persona 1.

Figura 31: Trasnformacion inversa del sistema polar de la persona 1.

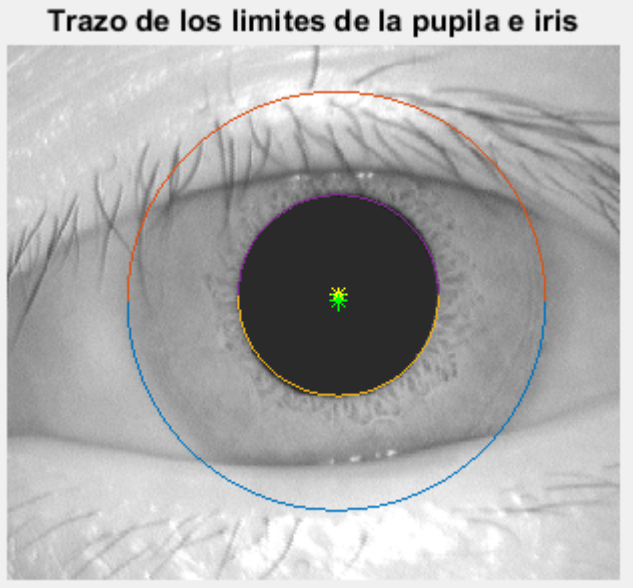


Figura 32: Trazo de los limites de la pupila e iris de la persona 2.

Figura 33: Trasnformación del iris normalizado y ecualizado a un sistema polar de la persona 2.

Figura 34: Trasnformación inversa de un sistema polar de la persona 2.

4.4.1. Pruebas

En las figuras 32,33 y 34 se muestran los resultados obtenidos con la imagen de la persona 2.

En las figuras 35,36 y 37 se muestran los resultados obtenidos con la imagen de la persona 3.

En las figuras 38,39 y 40 se muestran los resultados obtenidos con la imagen de la persona 4.

En las figuras 41,**??** y 43 se muestran los resultados obtenidos con la imagen de la persona 5.

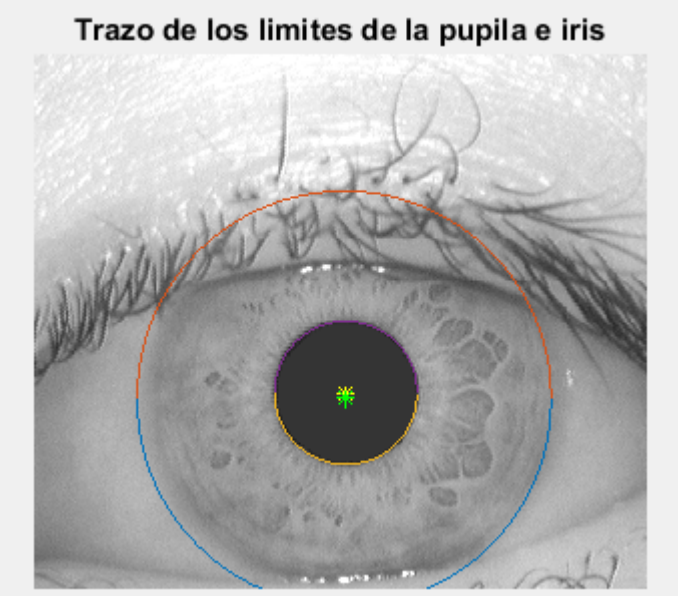


Figura 35: Trazo de los limites de la pupila de iris de la persona 3.

Figura 36: Transformación del iris normalizado y ecualizadp a un sistema polar de la persona 3.

Figura 37: Transformación inversa del sistema polar de la persona 3.

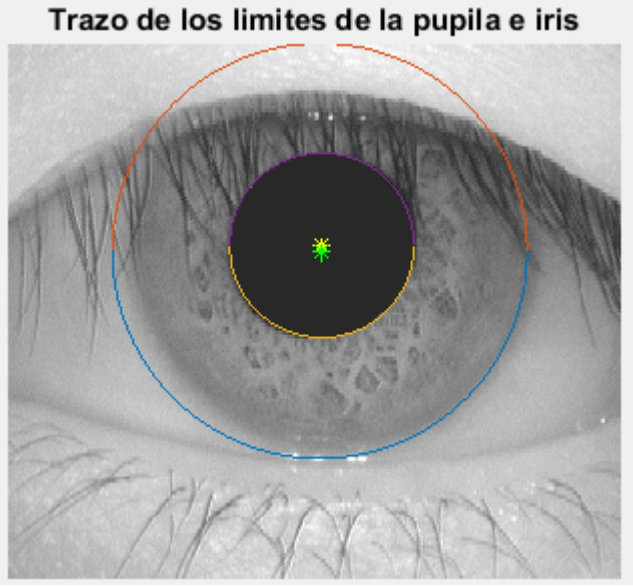


Figura 38: Trazo de los limites de la pupila e iris de la persona 4.

Figura 39: Transformación del iris normalizado y ecualizado a un sistema polar de la persona 4.

Figura 40: Trasnformación inversa del sistema polar de la persona 4.

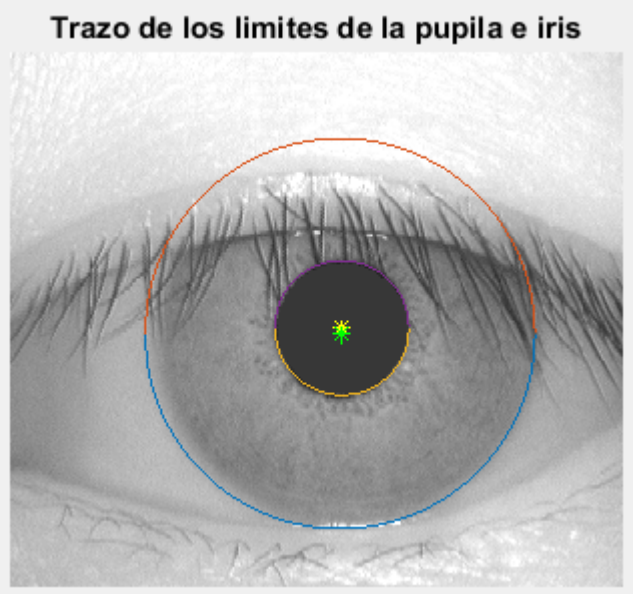


Figura 41: Trazo de los limites de la pupila e iris de la persona 5.

Figura 42: Transformación del iris normalizado y ecualizado a un sistema polar de la persona 5.

Figura 43: Trasnformación inversa del sistema polar de la persona 5.

5. Conclusiones

Se obtuvo un sistema de detección de iris, en el cual se realizaron varias etapas, las cuales se muestran en la sección de desarrollo, se encuentran los radiós del iris y de la pupila, y se extrae la banda comprendia entre dichos radios, además se transforma a un sistema normalizado. Se muestran los resultados de cada una de las etapas, así como también se presenta la descripción de cada una de las ellas.

Se comprendieron y utilizáron algunos de los algoritmos vistos y desarrollados en clases, así mismo se programaron otros, como prueba de lo aprendido a lo largo del semestre, y de esta forma complementar al sistema desarrollado.

Se muestran diferentes pruebas en la subsección de Desarrollo, para validar el sistema desarrollado y puede verse que el sistema cumple con los objetivos establecidos.

En los anexos se incluyen cada uno de los códigos implementados en cada una de las etapas, los cuales se desarrollaron en forma de funciones para facilitar su implementación.

Referencias

[1] Evanny Obregón Gamarra, Renato Oviedo Frasson, and Guillermo Kemper Vásquez. Biometría optica de iris. *Universidad Tecnológica de Pereira, Colombia*, 2010.

[2] Juan Alberto Devincenzi, María Laura Finamore, and Marcelo Naiouf2 Franco Chichizola and, Laura De Giusti. Reconocimiento biométrico de iris en ambientes de alta seguridad. *Facultad de Informática, UNLP*, 2012.

[3] Francesc Serratosa. Reconocimiento de personas por el iris. *Universidad Oberta de Cataluya*, 2013.

6. Anexos

6.1. Codigo principal

```
clc
clear all
close all
Im=imread('C:\Users\belen\Documents\Belen\rosebet\CASIA1\CASIA Iris Image Database
[Y X Z]=size(Im);
Conversion de la imagen original a escala de grises
if Z==3
    im=rgb2gray(Im);
```

```
10      figure(1);
11      subplot(1,3,1); imshow(im);
12      title('Imagen␣original');
13  else
14      im=Im;
15      figure(1);
16      subplot(1,3,1); imshow(im);
17      title('Imagen␣original');
18  end
19  %%
20  %%%%%%%%%%%%%%%%%%%%%%%%%%%%
21  %%Encontrar la pupila
22  %%%%%%%%%%%%%%%%%%%%%%%%%%%%
23  %filtro de la mediana para la imagen original
24  im_fil=filtro_mediana(im,X,Y);
25  figure(1);
26  subplot(1,3,2);imshow(im_fil);
27  title('Imagen␣con␣filtro␣de␣la␣mediana');
28  %Histrograma de la original
29  figure(2);
30  [h,x]=imhist(im);
31  subplot(3,1,1); plot(h);
32  title('Histograma␣de␣la␣imagen␣original');
33  %Histrograma de la filtrada
34  [h,x]=imhist(im_fil);
35  subplot(3,1,2); plot(h);
36  title('Histograma␣de␣la␣imagen␣con␣filtro␣de␣la␣mediana');
37  %Normalizacion de la imagen original
38  [imN_mB]=normalizar(im);
39  figure(1);
40  subplot(1,3,3);imshow(imN_mB);
41  title('Imagen␣original␣normalizada');
42  %Hitograma de la imagen original normalizada
43  [h_imN,x_imN]=imhist(imN_mB);
44  figure(2)
45  subplot(3,1,3); plot(h_imN);
46  title('Histograma␣de␣la␣imagen␣original␣normalizada');
47  %Normalizacion de la imagen filtrada
48 im_N=normalizar(im_fil);
49  figure(3);
50  subplot(1,2,1);imshow(im_N);
51  title('Imagen␣con␣filtro␣normalizada');
```

```
52 %histrograma de la imagen filtrada normalizada
53 [h,x]=imhist(im_N);
54 subplot(1,2,2); plot(h);
55 title('Histograma de la imagen con filtro normalizada');
56 %Obtencion del umbral de la imagen filtrada con normalizacion
57 T_umbral=umbral(h,x);
58 %Filtrado de la imagen usando el valor del umbral
59 imb=((uint8(im_N)-T_umbral)*255)/255;
60 figure(4);
61 subplot(2,1,1);imshow(double(imb),[]);
62 title('Imagen filtrada con el valor T_umbral');
63 %Ontercambiar los colores de imb
64 im_inter=1-imb;
65 subplot(2,1,2);imshow(im_inter,[]);
66 title('Intercambio de colores de los objetos con T_umbral ');
67 %Etiquetar los objetos que quedaron despues de la unbralizacion
68 [L_p labels im_blanc]=etiquetado(im_inter);
69 figure(5);
70 subplot(2,1,1); imshow(im_blanc,[]);
71 title('Binarizacion de la imagen con T_umbral');
72 subplot(2,1,2); imshow(L_p);
73 title('Etiquetado de la imagen con T_umbral');
74 %Encontrar la etiqueta con objeto de area maxima
75 [label_Amax vector]=area(labels,L_p,X,Y)
76 %Filtro para encontrar la pupila
77 im_pupila=find_pupila(Y,X,label_Amax,L_p);
78 figure(6); imshow(im_pupila)
79 title('Imagen de pupila extraida con centro en (xc,yc)');
80 hold on
81 %Enctontrar el centro de la pupila
82 xc=0;
83 yc=0;
84 stats=regionprops(im_pupila,'Centroid');
85 xc=stats.Centroid(1,1);
86 yc=stats.Centroid(1,2);
87 plot(xc,yc,'r*')
88 legend('Centro');
89 %Encontrar el radio de la pupila
90 bandaC=0;
91 r_pupila=find_radio(im_pupila,X,Y,xc,yc,bandaC);
92 %Obetener los datos para dibujar el circulo de la pupila
93 [xcV,ycV,ycnV]=draw_circle(r_pupila);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%Encontrar el iris
%%%%%%%%%%%%%%%%%%%%%%%%%%%%
im_iris=double(im_fil);
%Aplicacion de un operador para encontrar los bordes principales de un ojo
I_bordP=find_contornos(X,Y,im_iris);
figure(7);
subplot(2,1,1); imshow(I_bordP);
title('Extraccion de bordes de la imagen original');
%Encontrar el ruido de la imagen orignal
ruido=find_ruido(Im,Y,X);
figure(8);  imshow(ruido);
title('Extraccion del ruido de la imagen original');
%Elinar el ruido de los bordes obtenidos
im_sinR=eliminar_ruido(ruido,I_bordP,Y,X);
figure(7);
subplot(2,1,2); imshow(im_sinR);
title('Imagen de los bordes sin ruido');
%Aplicacion de operacion morfologica de Top-Hat
[im_o2 im_o1 im_c]=close_open(im_sinR);
figure(9);
subplot(1,3,1); imshow(im_c);
title('Imagen con cierre con se=1');
subplot(1,3,2); imshow(im_o1);
title('Imagen con apertura con se=1');
subplot(1,3,3); imshow(im_o2);
title('Imagen con apertura con se=2');
%Obtener solo una parte del ojo
[rec_ojo]=ojo_rectangulo(xc,yc,X,Y,im_o2,r_pupila);
figure(10);
subplot(1,2,1);imshow(rec_ojo);
title('Imagen recortada');
%Etiquetar los objetos restantes para enocntrar los de mayor area
im_copy=1-rec_ojo;
[L_I labels_I im_blancI]=etiquetado(im_copy);
subplot(1,2,2); imshow(L_I);
title('Etiquetado de los objetos encontrados');
[p_1 vec_I]=area (labels_I,L_I,X,Y);
vec_I(1,p_1)=0;
[maxAI, p_2]=max(vec_I);
%Encontrar los bordes del Iris
[im_iris]=find_iris(L_I,p_1,p_2,X,Y);
```

```
figure(11); imshow(im_iris);
title('Imagen de los bordes del iris');
%Encontrar el radio del iris
bandaC=10;
[r_iris]=find_radio(im_iris,X,Y,xc,yc,bandaC);
[xcVE,ycVE,ycnVE]=draw_circle(r_iris);
%Dibujar el circulo de la pupila y del iris
%Por simplicidad se asume que el vertical del iris y de la pupila coinciden
%pero el eje horizontal de la pupila esta dezplazado
ycE=yc+3;
figure(12);
subplot(2,1,1);imshow(Im);
hold on
plot(xcVE+xc-1,ycVE+ycE)
hold on
plot(xcVE+xc-1,ycnVE+ycE)
hold on
plot(xcV+xc,ycV+yc)
hold on
plot(xcV+xc,ycnV+yc)
hold on
plot(xc,yc,'*y');
hold on
plot(xc,ycE,'*g');
%legend('Centro Pupila','Centro Iris');
title('Trazo de los limites de la pupila e iris');
%Extraccion de la banda del iris con la imagen original normalizada
[mask_banda2]=extraccion_banda(Im,xc,yc,r_pupila,r_iris,ycE,X,Y);
subplot(2,1,2);  imshow(mask_banda2,[]);
title('Region del iris de la imagen original ');

[Y X Z]=size(mask_banda2);

if Z==3
    im_mB=rgb2gray(mask_banda2);
else
    im_mB=mask_banda2;
end

%Ecualizacion de la region del iris sin normalizar
ecualizada=ecualizar(im_mB,Y,X);
figure(14);
```

```
178 subplot(2,1,1);imshow(uint8(im_mB));
179 title('Region del iris original sin ecualizar');
180 subplot(2,1,2);imshow(uint8(ecualizada));
181 title('Region del iris original ecualizada');
182 %Extraccion de la region del iris normalizada
183 [mask_banda]=extraccion_banda(imN_mB,xc,yc,r_pupila,r_iris,ycE,X,Y);
184 figure(15);
185 subplot(2,1,1);  imshow(mask_banda,[]);
186 title('Region del iris normalizada sin ecualizar');
187 [Y, X Z]=size(mask_banda2);
188 if Z==3
189     im_mB=rgb2gray(mask_banda2);
190 else
191
192     im_mB=mask_banda2;
193 end
194 %Ecualizacion de la region del iris normalizado
195 ec_imN=ecualizar(im_mB,Y,X);
196 subplot(2,1,2);imshow(uint8(ecualizada));
197 title('Region del iris normalizada ecualizada');
198 %%
199 %Transformacion de la region del iris sin normalizar ni ecualizar
200 %a un sistema de coordenadas polares
201 [im_Tfil]=trans_PolRec(mask_banda2,X,Y,xc,ycE,r_pupila,r_iris);
202 figure(16);
203 subplot(2,1,1);imshow(im_Tfil,[]);
204 title('Transformacion del iris original a un sistema polar');
205 [x_i y_i]=size(im_Tfil);
206 [im_Tin]=trans_inversa(x_i,y_i,im,r_pupila,r_iris,xc,ycE);
207  figure(17);
208  subplot(2,1,1);imshow(im_Tin,[]);
209 title('Transformacion inversa del sistema polar');
210
211 %Transformacion de la region del iris con normalizacion y ecualizacion
212 %de a un sistema de coordenadas polares
213 [im_Tfil]=trans_PolRec(ec_imN,X,Y,xc,ycE,r_pupila,r_iris);
214 figure(16);
215 subplot(2,1,2);imshow(im_Tfil,[]);
216 title('Transformacion del iris normalizado y ecualizado a un sistema polar');
217 [x_i y_i]=size(im_Tfil);
218 [im_Tin]=trans_inversa(x_i,y_i,ec_imN,r_pupila,r_iris,xc,ycE);
219  figure(17);
```

```
220  subplot(2,1,2);imshow(im_Tin,[]);
221 title('Transformacion␣inversa␣del␣sistema␣polar');
```

6.1.1. Filtro de la mediana

```
1  function [im1] = filtro_mediana(im,X,Y)
2      k=0;
3      tam=11;
4      vec=zeros(1,tam);
5      res=tam/2-.5;
6      res2=res;
7      res3=1;
8      im1=im;
9      for i=1:Y
10         for j=1:X-res
11             if j>res
12                 for k=1:tam
13                     if res2==0
14                         vec(1,k)=im(i,j);
15                         res2=res+1;
16                     else if res2<=res && res2>0
17                         vec(1,k)=im(i,j-res2);
18                         res2=res2-1;
19                         else
20                         vec(1,k)=im(i,j+res3);
21                         res3=res3+1;
22                         end
23                     end
24                 end
25                 res2=res;
26                 res3=1;
27                 im1(i,j)=median(vec);
28             end
29         end
30     end
31
32 end
```

6.1.2. Algoritmo de normalización

```
1  function [imN]=normalizar(im1)
2      m=min(min(im1));
3      M=max(max(im1));
```

```
    imN=(im1-m)*(255/(M-m));
end
```

6.1.3. Algoritmo de detección automática de umbral

```
function [im1] = filtro_mediana(im,X,Y)
    k=0;
    tam=11;
    vec=zeros(1,tam);
    res=tam/2-.5;
    res2=res;
    res3=1;
    im1=im;
    for i=1:Y
        for j=1:X-res
            if j>res
                for k=1:tam
                    if res2==0
                        vec(1,k)=im(i,j);
                        res2=res+1;
                    else if res2<=res && res2>0
                        vec(1,k)=im(i,j-res2);
                        res2=res2-1;
                        else
                        vec(1,k)=im(i,j+res3);
                        res3=res3+1;
                        end
                    end
                end
                res2=res;
                res3=1;
                im1(i,j)=median(vec);
            end
        end
    end

end
```

6.1.4. Algoritmo de etiquetado

```
function [L etiqueta im2]=etiquetado(im_e)
    im_e=double(im_e);
    im2=im2bw(im_e);
```

```
    L = bwlabel(im2);
    etiqueta=max(max(L));
end
```

6.1.5. Algoritmo de detección de área

```
function [pos,vecEt]=area (etiqueta,L,X,Y)
    vecEt=ones(1,etiqueta);
    for t=1:etiqueta
        for i=1:Y
            for j=1:X
                if L(i,j)==t
                    vecEt(1,t)=vecEt(1,t)+1;
                end
            end
        end
    end
    %vecEt;
    [maxA, pos]=max(vecEt);
end
```

6.1.6. Algoritmo para encontrar la pupila

```
function [pupila]=find_pupila(Y,X,pos,L)
    pupila=zeros(Y,X);
    for i=1:Y
        for j=1:X
            if L(i,j)==pos
                pupila(i,j)=1;
            end
        end
    end

end
```

6.1.7. Algoritmo para encontrar el radio de la pupila y del iris

```
function [radio]=find_radio(L_p,X,Y,xc,yc,banda)
xmin=0;
xmax=0;
ymin=0;
ymax=0;
inicio=0;
for i=1:Y
```

```
8       for j=1:X
9           if L_p(i,j)==1
10              if inicio==0
11                  inicio=1;
12                  xauxm=j;
13                  yauxm=i;
14                  xauxM=j;
15                  yauxM=i;
16              else
17                  if j<xauxm
18                      xmin=j;
19                      xauxm=xmin;
20                  end
21                  if j>xauxM
22                      xmax=j;
23                      xauxM=xmax;
24                  end
25                  if i<yauxm
26                      ymin=i;
27                      yauxm=ymin;
28                  else
29                      ymin=yauxm;
30                  end
31                  if i>yauxM
32                      ymax=i;
33                      yauxM=ymax;
34                  end
35              end
36          end
37      end
38  end
39  % xmin
40  % xmax
41  % ymin
42  % ymax
43  rxm=xc-xmin-banda;
44  rxM=xmax-xc-banda;
45  rym=yc-ymin;
46  ryM=ymax-yc;
47  radios=[rxm rxM rym ryM];
48  radio=max(radios);
49  %r=radio;
```

```
50 end
```

6.1.8. Algoritmo para encontrar los límites de la pupila y del iris

```
function [xcV,ycV,ycnV]=draw_circle(radio)
    xcV=[-radio:0];
    tam=size(xcV);
    for i=1:tam(1,2)-1
        xcV(1,tam(1,2)+i)=(-1)*xcV(1,tam(1,2)-i);
    end
    ycV=sqrt(radio^2-xcV.^2);
    ycnV=-sqrt(radio^2-xcV.^2);
end
```

6.1.9. Algoritmo para detección de bordes utilizando el operador de Sobel

```
function [I_b]=find_contornos(x,y,im_I)
    % sx=[-1 0 0;0 1 0; 0 0 0] %roberts
    % sy=[0 0 -1;0 1 0; 0 0 0] ;

    % sx=[-1 0 1;-1 0 1; -1 0 1] %prewitt
    % sy=-sx';

    sx=[1 0 -1;2 0 -2; 1 0 -1]; %sobel
    sy=-sx;
    %sx=[-1 0 1;-2 0 2; -1 0 1] %sobel
    %sy=sx';

    % sx= [1,0,-1] %gradiente
    % sy= [1;0;-1]
    %
    gx=conv2(im_I,sx);
    gy=conv2(im_I,sy);

    gx1=gx(1:y,1:x);
    gy1=gy(1:y,1:x);
    g=sqrt((gx1.*gx1)+(gy1.*gy1));


    % gx=gx(:,1:x); %gradiente
    % gy=gy(1:y,:);
    % g=sqrt((gx.*gx)+(gy.*gy));

```

```
28      % a=atan(gy./gx);
29      a=atan(gy1./gx1);
30
31      figure(13);
32      subplot(2,2,1); imshow(gx1,[]);
33      title('Gradiente en x de la imagen');
34      subplot(2,2,2); imshow(gy1,[]);
35      title('Gradiente en y de la imagen');
36      % subplot(2,2,3);imshow(gx,[]);
37      % subplot(2,2,4); imshow(gy,[]);
38      subplot(2,2,3); imshow(g,[]);
39      title('Magnitud del gradiente de la imagen');
40      subplot(2,2,4); imshow(a,[]);
41      title('Angulo del gradiente de la imagen');
42
43      I_b=zeros(y,x);
44      %figure; imshow(I);
45      for i=1:y
46          for j=1:x
47              if g(i,j)<15
48                  I_b(i,j)=1;
49              end
50          end
51      end
52
53  end
```

6.1.10. Algoritmo de para obtener el ruido de la imagen

```
1   function [newim]=find_ruido(Imutil,Y,X)
2       [h1,x1]=imhist(Imutil);
3       figure; stem(h1);
4       sum=0;
5       tam=size(x1);
6       for i=1:tam(1,1)
7           sum=h1(i,1)+sum;
8       end
9       sum;
10      prom=sum/tam(1,1);
11
12      ltam=1;
13      newvec=zeros(ltam,1);
14      for i=1:tam(1,1)
```

```
15          if h1(i,1)>=prom
16              newvec(ltam,1)=i;
17              ltam=ltam+1;
18          end
19      end
20      newvec;
21      newim=ones(Y,X);
22      for k=1:ltam-1
23          for i=1:Y
24              for j=1:X
25                  if Imutil(i,j)==newvec(k,1)
26                      newim(i,j)=0;
27                  end
28              end
29          end
30      end
31  end
```

6.1.11. Algoritmo de eliminación de ruido

```
1  function [newIM]=eliminar_ruido(im_ruido,newIM,Y,X)
2      for i=1:Y
3          for j=1:X
4              if im_ruido(i,j)==1
5                  newIM(i,j)=255;
6              end
7          end
8      end
9  end
```

6.1.12. Algoritmo Top-Hat con doble apertura

```
1  function [imedaux ime imed]=close_open(im_mul)
2      se=strel('disk',1);
3      ime=imerode(im_mul,se);
4      %figure; imshow(ime);
5
6      imed=imdilate(ime,se);
7      %figure; imshow(imed);
8
9      se=strel('disk',2);
10     imedaux=imdilate(imed,se);
11     %figure; imshow(imedaux);
12 end
```

6.1.13. Algoritmo de recorte

```
function [copy]=ojo_rectangulo(xc,yc,X,Y,imedaux,radio)
    xcp =round(xc);
    ycp = round(yc);
    largo=120;
    r_p=round(radio)+5;
    xini=xcp-largo;
    xfin=xcp+largo;
    yini=ycp-r_p;
    yfin=ycp+r_p;
    %Eliminar los objetos fuera de los limites y_ini e y_fin
    %y de los x_ini e x_fin.
    copy=ones(Y,X);
    for i=1:Y
        if i>yini && i<yfin
            for j=1:X
                if j>xini && j<xfin
                    copy(i,j)=imedaux(i,j);
                end
            end
        end
    end
    %figure; imshow(copy);
    largo=r_p+30;
    xini=xc-largo;
    xfin=xc+largo;
    yini=yc-r_p;
    yfin=yc+r_p;
    %Eliminar los objetos dentro de los limites y_ini e y_fin
    %y de los x_ini e x_fin.
    for i=1:Y
        if i>yini && i<yfin
            for j=1:X
                if j>xini && j<xfin
                    copy(i,j)=1;
                end
            end
        end
    end
    %figure; imshow(copy);
end
```

6.1.14. Algoritmo para encontrar los bordes del iris

```
function [iris]=find_iris(L,pos1,pos2,X,Y)
    iris=zeros(Y,X);
    for i=1:Y
        for j=1:X
            if L(i,j)==pos1 || L(i,j)==pos2
                iris(i,j)=1;
            end
        end
    end
end
```

6.1.15. Algoritmo para extraer la región del iris

```
function [mascara]=extraccion_banda(Im_O,xc,yc,r,rE,ycE,X,Y)
    mascara=zeros(Y,X);
    for i=1:Y
        for j=1:X
            if ((i-(ycE))^2+(j-xc)^2)<rE^2 && ((i-yc)^2+(j-xc)^2)>r^2
                mascara(i,j)=Im_O(i,j);
%             else
%                 mascara(i,j)=255;
            end
        end
    end
end
```

6.1.16. Algoritmo para ecualizar

```
function [newx]=ecualizar(im_mB,m,n)

    %x es un vector o matriz formado por enteros

    im_mB=double(im_mB);
    mayor=max(max(im_mB));
    menor=min(min(im_mB));

    if menor>=0

    histograma=zeros(1,mayor);

    for i=1:m
        for j=1:n
```

```
15              for k=1:length(histograma)
16                  if k==im_mB(i,j)
17               histograma(k)=histograma(k)+1;
18                  end;
19              end;
20          end;
21      end;
22
23      else
24         mayor2=0-menor+1;
25         histograma=zeros(1,mayor2+mayor);
26
27      for i=1:m
28          for j=1:n
29              for k=menor:mayor
30                  if k==im_mB(i,j)
31                   histograma(k+mayor2)=histograma(k+mayor2)+1;
32                  end;
33              end;
34          end;
35      end;
36
37      end;
38
39      histograma=histograma/(n*m);
40      %Normalized histogram
41
42      ecu=zeros(1,length(histograma));
43
44      for i=1:length(histograma)
45          for j=1:i
46              ecu(i)=ecu(i)+histograma(j);
47              %ecu has the acumulative histogram
48          end;
49      end;
50
51
52      if menor>=0
53
54      for i=1:m
55          for j=1:n
56              if im_mB(i,j)~=0
```

```
57              newx(i,j)=ecu(im_mB(i,j))*(mayor-menor)+menor;
58               else
59               newx(i,j)=ecu(im_mB(i,j)+1)*(mayor-menor)+menor;
60               end;
61          end;
62      end;
63
64      else
65
66      for i=1:m
67          for j=1:n
68              newx(i,j)=ecu(im_mB(i,j)+mayor2)*(mayor-menor)+menor;
69          end;
70      end;
71
72
73      end;
74      newx=round(newx);
75  end
```

6.1.17. Algoritmo para transformación de coordenadas

```
1  function [V]=trans_PolRec(mask_banda3,X,Y,xcx,ycx,rp,ri)
2      rp=round(rp)+1;
3      ri=round(ri)+1;
4      N=ri-rp;
5      y_lon=[1:N];
6      M=360;
7      x_lon=[1:M];
8      for i=1:Y
9          for j=1:X
10             rn=sqrt((j-xcx)^2+(i-ycx)^2);
11             alfa=atand((i-ycx)/(j-xcx));
12             if alfa<0
13                 alfa=90+alfa;
14             else
15                 alfa=alfa;
16             end
17             if j>xcx && i>ycx
18                 alfa=270-alfa;
19             else
20                 if j<xcx && i<ycx
21                     alfa=90-alfa;
```

```
22                  else
23                      if j<xcx && i>ycx
24                          alfa=180-alfa;
25                      else
26                          if j>xcx && i<ycx
27                              alfa=360-alfa;
28                          end
29                      end
30                  end
31              end
32
33              if rn>=rp && rn<=ri
34                  xp=round(((M)*(alfa))/(2*180));
35                  yp=round(((N)*(rn-rp))/(ri-rp));
36                  V(yp+1,xp+1)=mask_banda3(i,j);
37              end
38          end
39      end
40
41  end
```

6.1.18. Algoritmo para transformación inversa

```
1  function [im_trans]=trans_inversa(X,Y,I_o,rp,ri,xo,yo)
2      for i=1:Y
3          for j=1:X
4              alfa=(((2*pi*i)/(X))-pi)*(180/(2*pi));
5              rn=rp+(((ri-rp)*i)/Y);
6              x=rn*sind(alfa)+xo;
7              y=rn*cosd(alfa)+yo;
8              im_trans(j,i)=I_o(round(x),round(y));
9          end
10     end
11
12 end
```