UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS
FACULTAD INGENIERÍA ELECTRÓNICA
E.A.P. DE INGENIERIA ELECTRÓNICA

Reconocimiento Biométrico de Huellas Dactilares y su implementación en DSP

TESIS para optar el Título profesional de: INGENIERO ELECTRÓNICO

Jorge Leonid Aching Samatelo
David Augusto Rojas Vigo
ASESORES: Ing. Flavio Carrillo Gomero

LIMA – PERÚ 2005

A mis padres César y Marlene por todo el apoyo y cariño brindado, a mis abuelos Ursula que partió tempranamente y Domingo por tanta paciencia y enseñanzas de toda una vida; a mis abuelos Jorge y Enith por su amor no obstante la distancia; a Gabriela por compartir su historia con la mía.

Jorge Leonid Aching Samatelo

A mis amados padres: Carmen y Anatolio, por todo el apoyo, paciencia y amor que siempre me brindan, pero sobretodo por ser mi inspiración y la fuente de mi más sincero respeto y orgullo. Son el más grande tesoro en mi vida.

David Augusto Rojas Vigo

Agradecimientos

A nuestros padres por el por el infinito amor que nos prodigan, por sus sacrificios y desvelos para educarnos y hacer de nosotros profesionales al servicio del país.

Al Ing. Flavio Carrillo Gomero, asesor en la elaboración de la tesis, por su dedicación y juiciosa dirección, por el soporte puesto al servicio del presente trabajo. Al Ing. Guillermo Tejada Muñoz, por su confianza, el constante estímulo y asistencia para nuestra labor de investigación a través del Instituto de Investigación y el Laboratorio de la facultad, con material bibliográfico, equipos de cómputo y medición. Al Ing. Ever Cifuentes Noreña, por el desinteresado apoyo que nos ha brindado para seguir desarrollándonos profesionalmente.

A Ramón, Luis y Wilfredo amigos del grupo de Simulación Humana de la UNI. Finalmente agradecemos a nuestros familiares, amigos y profesores, que participaron directa o indirectamente en el desarrollo de este trabajo.

Lista de Símbolos

$\mathbf{T}_S$	Plantilla de la impresión dactilar almacenada en la base de datos, también conocida como impresión dactilar secundaria.
$\mathbf{T}_P$	Plantilla de la impresión dactilar adquirida para la verificación, también conocida como impresión dactilar primaria.
$\mathbf{I}$	Imagen digital definida como una matriz de tamaño $N^I_{Fil} x N^I_{Col}$, donde, $\mathbf{I}(u,v)$ representa la intensidad del píxel de la u-ésima fila y la v-ésima columna.
$\mathbf{I}_{Mej}$	Imagen mejorada de $\mathbf{I}$, definida como una matriz de tamaño $N^I_{Fil} x N^I_{Col}$, donde, $\mathbf{I}_{Mej}(u,v)$, representa la intensidad modificada (mejorada) del píxel (u,v).
$\mathbf{I}_{Bin}$	Imagen binarizada de $\mathbf{I}$, definida como una matriz de tamaño $N^I_{Fil} x N^I_{Col}$, donde $\mathbf{I}_{Bin}(u,v)$ representa el valor binario asumido por el píxel (u,v).
$\mathbf{I}_{esq}$	Imagen esqueletizada de $\mathbf{I}_{Bin}$, definida como una matriz de tamaño $N^I_{Fil} x N^I_{Col}$, donde $\mathbf{I}_{esq}(u,v)$ representa el valor binario asumido por el píxel (u,v) perteneciente a una cresta adelgazada o al fondo de la imagen.
$\mathbf{S}^k_{ij}$	Bloque (i,j)-ésimo vinculado a $\mathbf{I}$ de tamaño $N_k x N_k$.
$\mathbf{S}^{N_B,N_{TR}}_{ij}$	Bloque (i,j)-ésimo vinculado a $\mathbf{I}$ de tamaño $N_B x N_B$ y con un traslapamiento de N_{TR} píxeles.
$\frac{\partial \mathbf{I}(u,v)}{\partial x}, \frac{\partial \mathbf{I}(u,v)}{\partial y}$	Derivadas parciales de primer orden, respecto a los ejes coordenados x e y.
$\frac{\partial^2 \mathbf{I}(u,v)}{\partial x^2}, \frac{\partial^2 \mathbf{I}(u,v)}{\partial y^2}, \frac{\partial^2 \mathbf{I}(u,v)}{\partial x \partial y}$	Derivadas parciales de segundo orden, respecto a los

	ejes coordenados x e y.
$\lceil \bullet \rceil$	Operador que define el menor entero mayor que $\bullet$.
$\lfloor \bullet \rfloor$	Operador que define el mayor entero menor que $\bullet$.
$\bullet \% \bullet$	Operador binario que determina el resto de la división del operando izquierdo entre el operando derecho, comunmente denominado módulo.
$\|\bullet\|$	Operador módulo de un número complejo o vector.
$\lvert\bullet\rvert$	Operador valor absoluto.
$\|\nabla \mathbf{I}(u,v)\|$	Magnitud del Vector Gradiente en el píxel (u,v).
$\theta_{\nabla}(u,v)$	Dirección del Vector Gradiente en el píxel (u,v).
$\mathbf{G}_x$ y $\mathbf{G}_y$	Componentes del Vector Gradiente de la imagen $\mathbf{I}$, definida como matrices de tamaño $N^I_{Fil} x N^I_{Col}$, donde $\mathbf{G}_x(u,v)$ y $\mathbf{G}_y(u,v)$ indican la variación en la intensidad de los niveles de grises segun las direcciones x e y analizada en la posición (u,v) de la imagen $\mathbf{I}$.
$\mathbf{G}_{xx}, \mathbf{G}_{yy}$ y $\mathbf{G}_{xy}$	Derivadas de segundo orden del Vector Gradiente, definidas como matrices de tamaño $N^I_{Fil} x N^I_{Col}$, donde $\mathbf{G}_{xx}(u,v), \mathbf{G}_{yy}(u,v)$ y $\mathbf{G}_{xy}(u,v)$ son las segundas derivadas parciales en las direcciones x e y analizada en la posición (u,v) de la imagen $\mathbf{I}$.
$\mathbf{G}_{s,x}$ y $\mathbf{G}_{s,y}$	Componentes del Vector Gradiente al cuadrado, definidas como matrices de tamaño $N^I_{Fil} x N^I_{Col}$, donde $\mathbf{G}_{s,x}(i,j)$ y $\mathbf{G}_{s,y}(i,j)$ los componentes real y imaginario de la representación compleja del Vector Gradiente analizada en el bloque $\mathbf{S}^k_{ij}$.
$\overline{\mathbf{G}}_{s,x}$ y $\overline{\mathbf{G}}_{s,y}$	Componentes promedio del Vector Gradiente al cuadrado vinculado a cada bloque $\mathbf{S}^k_{ij}$, definidas como matrices de tamaño $N_{Fil} x N_{Col}$, donde, $\overline{\mathbf{G}}_{s,x}(i,j)$ y $\overline{\mathbf{G}}_{s,y}(i,j)$ son los Vectores

	Gradiente promediados pertenecientes al bloque $\mathbf{S}_{ij}^{k}$.
$\mathbf{G}_{ij}^{xy}$	Agrupación de los Vectores Gradiente pertenecientes al bloque $\mathbf{S}_{ij}^{k}$, dentro de una matriz $N_k^2 x 2$.
$\mathbf{U}_{ij}$	Matriz ortogonal $N_k^2 x N_k^2$, tal que sus columnas contienen los eigenvectores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$.
$\mathbf{V}_{ij}$	Matriz ortogonal 2x2, tal que sus columnas contienen los eingenvectores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$.
$\mathbf{D}_{ij}$	Matriz diagonal $N_k^2 x 2$, cuyos elementos de la diagonal son las raíces cuadradas de los eingenvalores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$.
$\mathbf{\Phi}$	Dirección promedio del Vector Gradiente al cuadrado, definida como una matriz de tamaño $N_{Fil} x N_{Col}$, en donde $\mathbf{\Phi}(i,j)$ representa la orientación dominante en el bloque $\mathbf{S}_{ij}^{k}$.
γ_x y γ_y	Componentes sinusoidales de $\mathbf{\Phi}$, definidas como matrices de tamaño $N_{Fil} x N_{Col}$, donde $\gamma_x(i,j)$ y $\gamma_y(i,j)$ son el componente cosenoidal y senoidal de $2.\mathbf{\Phi}(i,j)$.
γ'_x y γ'_y	Componentes sinusoidales filtrados de $\mathbf{\Phi}$, definidas como matrices de tamaño $N_{Fil} x N_{Col}$, donde $\gamma'_x(i,j)$ y $\gamma'_y(i,j)$ son los resultados de filtrar γ_x y γ_y con la máscara $\mathbf{W}_{SH}$ en la posición correspondiente al bloque $\mathbf{S}_{ij}^{k}$.
Θ	Imagen de Orientación, definida como una matriz de tamaño $N_{Fil} x N_{Col}$, donde $\Theta(i,j)$ representa la dirección promedio normal a las crestas del bloque $\mathbf{S}_{ij}^{k}$.
Θ_X y Θ_Y	Componentes sinusoidales de Θ, definidas como matrices de tamaño $N_{Fil} x N_{Col}$, donde $\Theta_X(i,j)$ y $\Theta_Y(i,j)$ son los componentes cosenoidal y senoidal de $\Theta(i,j)$.
$\nabla\Theta_X(i,j)$ y $\nabla\Theta_Y(i,j)$	Vectores Gradiente de Θ_X y Θ_Y calculados en la posición (i,j).

$\|\nabla\Theta_{\mathbf{X}}(i,j)\|$	Magnitud del Vector Gradiente de $\Theta_{\mathbf{X}}$ calculada en la posición (i,j).
$\alpha_{\mathbf{XY}}(i,j)$	Seno del ángulo formado por los vectores normalizados $\nabla\Theta_{\mathbf{X}}(i,j)$ y $\nabla\Theta_{\mathbf{Y}}(i,j)$.
Θ_{ij}	Submatriz de Θ, definida como una matriz de tamaño qxq centrada en la posición (i,j) de la matriz Θ.
Θ_{ij}^{k}	k-ésima columna de la sub-matriz Θ_{ij}.
$\mathbf{u}_{ij}$	Vector promedio de las columnas de la sub matriz Θ_{ij}, definido como un vector de tamaño $qx1$.
Σ_{ij}	Matriz de Covarianza de Θ_{ij}, cuyo máximo eingenvalor viene denotado por λ_{ij}.
Ψ	Transformación de covarianza.
$\mathbf{B}_{\Psi}$	Binarización de la transformación de covarianza.
$\mathbf{W}_{WN}$	Máscara de filtro Winner Pasabajos bidimensional de tamaño $N_{W_{WN}} x N_{W_{WN}}$.
$\mathbf{W}_{GS}$	Máscara Gaussiana bidimensional de tamaño $N_{W_{GS}} x N_{W_{GS}}$.
$\mathbf{W}_{GB}$	Máscara de Gabor bidimensional de tamaño $N_{W_{GB}} x N_{W_{GB}}$.
$\mathbf{W}_{SH}$	Máscara de un filtro pasa bajos bidimensional de tamaño $N_{W_{SH}} x N_{W_{SH}}$.
$\mathbf{W}_{x}$ y $\mathbf{W}_{y}$	Máscaras a usar para obtener el Vector Gradiente, definidas como matrices de tamaño impar (recomendado 3x3).
$\mathbf{W}_{xx}$, $\mathbf{W}_{yy}$ y $\mathbf{W}_{xy}$	Máscaras equivalentes a las representaciones discretas de las derivadas paraciales $\frac{\partial^2 \mathbf{I}(u,v)}{\partial x^2}$, $\frac{\partial^2 \mathbf{I}(u,v)}{\partial y^2}$ y $\frac{\partial^2 \mathbf{I}(u,v)}{\partial x \partial y}$, definidas como matrices de tamaño 3x3.
$\mathbf{C}_{o}$	Matriz de Consistencia definida como una matriz de tamaño $N_{Fil} x N_{Col}$, donde $\mathbf{C}_{o}(i,j)$ es el error cuadrático medio de las diferencias angulares respecto al elemento central del

	bloque $\mathbf{S}_{ij}^{k}$.
$U_E(\theta_p(t,k))$	Función de energía; se define como la suma de las distancias normalizadas al cuadrado de cada uno de los vectores vinculados con las observaciones vecinas respecto al vector vinculado a θ_p.
$d_E(\theta_v, \theta_p)$	Distancia Euclidiana entre la observación θ_p con cada uno de sus vecinos θ_V.
$d_M(\theta_v, \theta_p)$	Distancia de Mahalanobis entre la observación θ_p con cada uno de sus vecinos θ_V.
$\Im$	Imagen de Frecuencias, definida como una matriz de tamaño $N_{Fil} x N_{Col}$, en donde $\Im(i,j)$ representa la frecuencia local de las crestas del bloque $\mathbf{S}_{ij}^{k}$.
Γ_{xx}, Γ_{yy}, Γ_{xy}, Γ_x y Γ_y	Valores ponderados de las respectivas derivadas calculados sobre una ventana de tamaño $N_\Im x N_\Im$ centrada en el píxel (u_{cent}, v_{cent}).
$\Re$	Imagen de Regiones, definida como una matriz de tamaño $N_{Fil}^{I} x N_{Col}^{I}$, donde $\Re(u,v)$ indica una categoría relacionada con el píxel $\mathbf{I}(u,v)$.
Mean$[\mathbf{I}(u,v)]$	Valor promedio local calculado sobre una ventana Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$ centrada en el píxel (u,v) de la imagen $\mathbf{I}$.
Mean$[\mathbf{I}^2(u,v)]$	Valor promedio local de $\mathbf{I}^2$ calculado sobre una ventana Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$ centrada en el píxel (u,v) de la imagen $\mathbf{I}$.
Var$[\mathbf{I}(u,v)]$	Varianza promedio local calculado sobre una ventana Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$ centrada en el píxel (u,v) de la imagen $\mathbf{I}$.
Ener$[\mathbf{I}(u,v)]$	Energia promedio local calculada sobre una ventana

	Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$ centrada en el píxel (u,v) de la imagen $\mathbf{I}$.
$Coh[\mathbf{I}(u,v)]$	Nivel de Coherencia calculado sobre una ventana Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$ centrada en el píxel (u,v) de la imagen $\mathbf{I}$.
$Irr[\mathbf{I}(u,v)]$	Nivel de Irregularidad calculado considerando el valor de $Coh[\mathbf{I}(u,v)]$ y $\Re(u,v)$.
$\mathbf{x}$	Observación asociada al píxel/bloque $(u,v)/(i,j)$ de la imagen $\mathbf{I}$.
$\Omega = \{w_{-1}\ ,\ w_1\}$	Clases a las que cada observación podrá pertenecer.
$S_{\mathbf{x}}$	Conjunto de entrenamiento inicial de $n_{\mathbf{x}}$ pares, compuesto por los subconjuntos $S_{\mathbf{x}_{-1}}$ y $S_{\mathbf{x}_1}$, tal que: $n_{\mathbf{x}} = n_{\mathbf{x}_{-1}} + n_{\mathbf{x}_1}$, donde $n_{\mathbf{x}_{-1}}$ y $n_{\mathbf{x}_1}$ son el número de observaciones de $S_{\mathbf{x}_{-1}}$ y $S_{\mathbf{x}_1}$, respectivamente.
$\mathbf{X}_k$	Observaciones realizadas, vinculadas al conjunto de entrenamiento $S_{\mathbf{x}_k}$. Es una matriz de tamaño $3xn_{\mathbf{x}_k}$, tal que $k = -1,1$.
$\mathbf{X}^i_{j:k}$	Observación del conjunto de entrenamiento $S_{\mathbf{x}_k}$ para la característica i-ésima, tal que $i = 1,2,3$ y $j = 1,2,\cdots,n_{\mathbf{x}_k}$.
$\mathbf{X}_{j:k}$	Observación j-ésima del conjunto de entrenamiento $S_{\mathbf{x}_k}$, es un vector columna 3x1 de la matriz $\mathbf{X}_k$.
$\mathbf{X}^i_k$	Valores en las observaciones del conjunto de entrenamiento $S_{\mathbf{x}_k}$ para la i-ésima característica, es un vector fila $1xn_{\mathbf{x}_k}$ de la matriz $\mathbf{X}_k$.
N_{int} y N_{out}	Número de nodos en la capa de entrada y de competición de una red SOFM respectivamente.
μ_{ij}	Peso vinculado a la conexión entre el nodo j-ésimo de la capa de entrada y el nodo i-ésimo de la capa de

	competición en una red SOFM.
$\mathbf{U}_i$	Vector de pesos vinculado a las conecciones del i-ésimo nodo de la capa de competición con cada uno de los N_{int} nodos de la capa de entrada en una red SOFM.
$\mathbf{N}_i$	Conjunto de vecindad alrededor del i-ésimo nodo de la capa de competición de una red SOFM.
$h_{ci}(t)$	Función de retroalimentación del nodo i al nodo ganador c de una red SOFM.
$\mathbf{y}_j$	Observación condensada j-ésima.
$\overline{\mathbf{y}}_j$	Vector aumentado referido a $\mathbf{y}_j$.
c_j	Clase verdadera asignada a la observacion $\mathbf{y}_j$.
$\tilde{c}_j$	Clase supuesta determinada a la observacion $\mathbf{y}_j$.
e_j	Error entre la clase calculada $\tilde{c}_j$ y la clase verdadera c_j.
$S_{\mathbf{Y}}$	Conjunto de aprendizaje para el paso de clasificación de $n_{\mathbf{Y}}$ pares, compuesto por los subconjuntos $S_{\mathbf{Y}_{-1}}$ y $S_{\mathbf{Y}_1}$, tal que: $n_{\mathbf{Y}} = n_{\mathbf{Y}_{-1}} + n_{\mathbf{Y}_1}$, donde $n_{\mathbf{Y}_{-1}}$ y $n_{\mathbf{Y}_1}$ son el número de observaciones de $S_{\mathbf{Y}_{-1}}$ y $S_{\mathbf{Y}_1}$, respectivamente.
$\mathbf{w}_p$	Vector de pesos del *Perceptron*.
$u_k(\mathbf{y}_j)$	Función de membresía que describe el grado en que la j-ésima observación del conjunto de entrenamiento $S_{\mathbf{Y}}$ pertenece a la k-ésima clase.
$d_k(\mathbf{y})$	Distancia entre la observación $\mathbf{y}$ y el vector promedio μ_k vinculado a la k-ésima clase.
$\mathbf{CG}(i, j)$	Magnitud del promedio del Vector Gradiente del bloque (i, j).
$\mathbf{CV}(i, j)$	Varianza de una vecindad orientada centrada en el bloque (i, j).

$\mathbf{I}_{rot}^{(i,j)}(u',v')$	Matriz de tamaño $N_{rot}xN_{rot}$ que contiene la vecindad local orientada respecto al bloque $\mathbf{S}_{ij}^{k}$.
$Mf_u[\bullet]$	Operador que calcula el valor promedio de la fila u-ésima de una matriz.
$Mc_v[\bullet]$	Operador que calcula el valor promedio de la columna v-ésima de una matriz.
$\mathbf{x}_1^{ctr} = [\mu_{CG}^1 \quad \mu_{CV}^1]^T$	Centroide representativo vinculado a la clase w_1.
$\mathbf{x}_{-1}^{ctr} = [\mu_{CG}^{-1} \quad \mu_{CV}^{-1}]^T$	Centroide representativo vinculado a la clase w_{-1}.
$\mathbf{C}_r$	Contorno de un cuadrado de $N_{\mathbf{C}_r}xN_{\mathbf{C}_r}$ elementos de Θ, tal que, el sub índice r establece el tamaño del cuadrado.
$\varepsilon_X(k)$ y $\varepsilon_Y(k)$	Coordenadas x e y de $\mathbf{C}_r$.
$FFT2\{\bullet\}$	Operador que define la Transformada Rápida de Fourier Bidimensional.
$IFFT2\{\bullet\}$	Operador que define la Transformada Inversa de Fourier Bidimensional.
$\mathbf{P}_{\Theta,\mathbf{C}}(i,j)$	Índice de Poincaré, definido como la suma algebraica de las diferencias de las orientaciones entre elementos adyacentes de la curva $\mathbf{C}_r$.
$\frac{\partial \mathbf{I}(u,v)}{\partial p}, \frac{\partial \mathbf{I}(u,v)}{\partial q}$	Derivadas parciales de primer orden calculadas para el píxel (u,v) perteneciente al bloque $\mathbf{S}_{ij}^{k}$, respecto al vector $\mathbf{v} = [\cos(\Theta(i,j)) \quad sen(\Theta(i,j))]^T$.
$\frac{\partial^2 \mathbf{I}(u,v)}{\partial p^2}, \frac{\partial^2 \mathbf{I}(u,v)}{\partial q^2}, \frac{\partial^2 \mathbf{I}(u,v)}{\partial p \partial q}$	Derivadas parciales direccionales de segundo orden calculadas para el píxel (u,v) perteneciente al bloque $\mathbf{S}_{ij}^{k}$, respecto al vector $\mathbf{v} = [\cos(\Theta(i,j)) \quad sen(\Theta(i,j))]^T$.
$\mathbf{D}^{(m,n)}(u,v)$	Función de interpolación basada en los polinomios de Chebyshev para la aproximación discreta de las derivadas parciales de orden superior respecto a los ejes coordenados,

	tal que m indica el orden de la derivada parcial respecto al eje x y n indica el orden de la derivada parcial respecto al eje y.
$POL_{(0,L)}(z), POL_{(1,L)}(z), POL_{(2,L)}(z)$	Polinomios de interpolación de Chebyshev.
$P_1, P_2, \cdots, P_8$	Píxeles del contorno de una máscara 3x3 centrada en el píxel P.
$N(P)$	Número de vecinos no cero de P.
$S(P)$	Número de transiciones de cero a uno, en la secuencia $P_1, P_2, \cdots, P_8$.
MF y MB	Conjuntos de Tripletas de N_{MF} y N_{MB} elementos respectivamente, que representan las posiciones y orientaciones de las minucias de finalización y bifurcación.
$CN(u, v)$	Definición de Rutovitz para el número de cruces por cero, calculado sobre el píxel (u, v) de la imagen $\mathbf{I}_{esq}$.
PF y PB	Conjuntos de pares de puntos de N_{MF} y N_{MB} elementos respectivamente, vinculados a las minucias de finalización y de bifurcación.
RF_i	Conjunto de pares ordenados de N_{RTS} elementos, que definen la ruta vinculada a la minucia de finalización i-ésima.
$\theta_{F,i}$	Orientación vinculada a la minucia de finalización i-ésima.
RB_{i1}, RB_{i2} y RB_{i3}	Conjuntos de pares ordenados de N_{RTS} elementos, que definen las tres rutas vinculadas a la minucia de bifurcación i-ésima.
$\theta_{B,i1}$, $\theta_{B,i2}$ y $\theta_{B,i3}$	Orientación vinculada a cada ruta de la minucia de bifurcación i-ésima.
$\theta_{B,i}$	Orientación vinculada a la minucia de bifurcación i-ésima.

$\mathbf{S}_i^{B,b,k}$	Sector i-ésimo en la región circular de B bandas concéntricas de b píxeles de ancho, tal que cada banda contiene k sectores circulares.
K_i	Número de píxeles en el sector $\mathbf{S}_i^{B,b,k}$.
V_i y M_i	Valor promedio y varianza estimada para el sector $\mathbf{S}_i^{B,b,k}$.
V_o y M_o	Valor promedio y varianza deseada para todo sector $\mathbf{S}_i^{B,b,k}$.
$\mathbf{NOR}_i(u,v)$	Valor normalizado del píxel (u,v) perteneciente al sector $\mathbf{S}_i^{B,b,k}$.
$\mathbf{F}_{i\theta}(u,v)$	Valor del píxel (u,v) perteneciente al sector $\mathbf{S}_i^{B,b,k}$ de la imagen filtrada en la dirección θ.
$\mu_{i\theta}$	Promedio de los valores $\mathbf{F}_{i\theta}(u,v)$.
$V_{i\theta}$	Desviación éstandar de los valores $\mathbf{F}_{i\theta}(u,v)$, define el elemento i-ésimo del *Finger Code* vinculado a la imagen filtrada en la dirección θ.
$V_{i\theta}^R$	Rotación por R pasos del elemento i-ésimo del *Finger Code* vinculado a la imagen filtrada en la dirección θ.
υ_A	Conjunto de N_A puntos minucias obtenidas por la técnica propuesta.
υ_R	Conjunto de N_R puntos minucias obtenidas por un experto.
GI	Índice de virtud.
$\mathrm{Aprd}_{(i,j)}$	Número de minucias apareadas en la ventana r-ésima.
$\mathrm{Cld}_{(i,j)}$	Factor de calidad del bloque (i,j).
$\mathrm{Dstr}_{(i,j)}$	Número de minucias destruidas en el bloque (i,j).
$\mathrm{Intr}_{(i,j)}$	Número de minucias insertadas en el bloque (i,j).
$\mathrm{Rls}_{(i,j)}$	Número de minucias reales en el bloque (i,j).

$\mathbf{I}_P$, $\mathbf{I}_S$	Impresiones dactilares primarias y secundarias.
$\mathbf{I}_P^{(\Delta i,\Delta j,\theta)}$	Rotación de $\mathbf{I}_P$ por un ángulo θ alrededor del origen (usualmente el centro de la imagen) y corrida $\Delta i, \Delta j$ píxeles en direcciones i, j respectivamente.
υ_P y υ_S	Vectores de características de $\mathbf{I}_P$ y $\mathbf{I}_S$ donde N_{MP} y N_{MS} denotan el número de elementos de υ_P y υ_S .
$\mathbf{M}$	Máscara a correlacionar sobre $\mathbf{I}$ de tamaño NxM, tal que: $\overline{N} = N/2$ y $\overline{M} = M/2$.
Mean[$\mathbf{M}$]	Valor promedio de $\mathbf{M}$.
Mean[$\mathbf{M}^2$]	Valor promedio de $\mathbf{M}^2$.
$\mathbf{M}_k$	Región local k-ésima extraída de $\mathbf{I}_S$, tal que $k = \{1, 2, \cdots, N_R\}$.
SSD($\mathbf{I}_S, \mathbf{I}_P$)	Suma de las diferencias cuadradas entre $\mathbf{I}_S$ y $\mathbf{I}_P$.
SSD[$\mathbf{I}(u, v), \mathbf{M}$]	Suma de las diferencias cuadradas entre la región extraída de $\mathbf{I}$ centrada en (u, v) y la máscara $\mathbf{M}$.
CC($\mathbf{I}_S, \mathbf{I}_P$)	Correlación cruzada estándar entre $\mathbf{I}_S$ y $\mathbf{I}_P$.
CC[$\mathbf{I}(u, v), \mathbf{M}$]	Correlación cruzada estándar entre la región extraída de $\mathbf{I}$ centrada en (u, v) y la máscara $\mathbf{M}$.
NCC[$\mathbf{I}(u, v), \mathbf{M}$]	Correlación cruzada normalizada entre la región extraída de $\mathbf{I}$ centrada en (u, v) y la máscara $\mathbf{M}$.
S($\mathbf{I}_S, \mathbf{I}_P$)	Similaridad entre $\mathbf{I}_S$ y $\mathbf{I}_P$.
SUM	Matriz de sumas acumuladas de tamaño $N_I x M_I$.
(u_k^p, v_k^p)	Posición k-ésima en la imagen $\mathbf{I}_P$ cuya vecindad produce la maxima correlación con la región $\mathbf{M}_k$ de la imagen $\mathbf{I}_S$.
(u_k^s, v_k^s)	Posición k-ésima en la imagen $\mathbf{I}_S$ que es el centro de la región $\mathbf{M}_k$.

POS_P y POS_S	Conjuntos de pares de puntos, que agrupan a todos los N_R pares (u_k^p, v_k^p) y (u_k^s, v_k^s) respectivamente.
m_i	Minucia i-ésima en $\mathbf{I}_P$.
m'_j	Minucia j-ésima en $\mathbf{I}_S$.
$sd(m'_j, m_i)$	Distancia espacial entre las posiciones de las minucias m_i y m'_j.
$dd(m'_j, m_i)$	Diferencia angular entre las orientaciones de las minucias m_i y m'_j.
map(•)	Función que mapea la minucia m'_j como m''_j de acuerdo con la transformación de similaridad dada.
mm(•)	Función que indica la coincidencia entre las minucias m''_j y m_i.
prd(i)	Función que determina el apareamiento entre las minucias de $\mathbf{I}_P$ y $\mathbf{I}_S$.
tr_i	Tripleta de puntos i-ésima que hace referencia al triángulo de Delaunay i-ésimo vinculado al conjunto de minucias υ_P.
tr'_j	Tripleta de puntos j-ésima que hace referencia al triángulo de Delaunay j-ésimo vinculado al conjunto de minucias υ_S.
ζ_P y ζ_S	Conjuntos de tripletas, que agrupan a todas las N_{TRP} y N_{TRS} tripletas tr_i y tr'_j respectivamente.
$\overline{tr}_i$	Tripleta de puntos i-ésima que hace referencia a un triángulo de Delaunay válido vinculado al conjunto de minucias υ_P.
$\overline{tr}'_j$	Tripleta de puntos j-ésima que hace referencia a un triángulo de Delaunay válido vinculado al conjunto de

	minucias υ_S.
$\overline{\zeta}_P$ y $\overline{\zeta}_S$	Nuevos conjuntos de tripletas, que agrupan a todas las N_{INVP} y N_{INVS} tripletas válidas $\overline{tr}_i$ y $\overline{tr}'_j$ respectivamente.
l_{i_1} , l_{i_2} y l_{i_3}	Longitud de los lados del triángulo vinculado a la tripleta $\overline{tr}_i$.
l'_{i_1} , l'_{i_2} y l'_{i_3}	Longitud de los lados del triángulo vinculado a la tripleta $\overline{tr}'_j$.
Q_i y Q'_j	Ángulos centrales vinculados a las tripletas $\overline{tr}_i$ y $\overline{tr}'_j$.
inv_i	Invarianza de triangulación i-ésima vinculada a la tripleta $\overline{tr}_i$.
inv_j	Invarianza de triangulación j-ésima vinculada a la tripleta $\overline{tr}'_j$.
ξ_P y ξ_S	Conjuntos de tripletas, que agrupan a todas las N_{INVP} y N_{INVS} invarianzas de triangulación inv_i y inv_j respectivamente.
IMth	Índice de similaridad entre υ_P y υ_S.
N_{MP}	Número de minucias de la imagen primaria.
N_{APRD}	Número de minucias apareadas o coincidentes.
$\Gamma\{\bullet\}$	Operador que define la acción de un filtro sobre la imagen de entrada.
$\mathbf{W}(u,v)$	Máscara de convolución espacial, definida como una matriz de tamaño NxM , teniendo como posición inicial el centro de la máscara por lo que los dominios para los indices son: $[-\overline{N},+\overline{N}]x[-\overline{M},+\overline{M}]$, tal que: $\overline{N}=\lfloor N/2 \rfloor$ y $\overline{M}=\lfloor M/2 \rfloor$.
$\mathbf{O}(u,v)=\mathbf{I}(u,v)*\mathbf{W}(u,v)$	Operación de convolución espacial entre la imagen de entrada $\mathbf{I}$ y la máscara $\mathbf{W}$, generandose la imagen de salida $\mathbf{O}$.

$W_{GS}(x, y; \sigma)$	Función de Gauss bidimensional, tal que, (x, y) son las coordenadas espaciales y σ es la desviación estándar de la función Gaussiana.
$W_{GB}(x, y; \sigma_x, \sigma_y, \lambda, \theta)$	Función de Gabor bidimensional, tal que, (x, y) son las coordenadas espaciales; σ_x y σ_y son las desviaciones estándares de las funciones Gaussianas; λ es la longitud de onda y θ es la orientación de la de la función.
$W_{GB}(x, y; \sigma, \gamma, \lambda, \theta)$	Definición alterna de la función de Gabor bidimensional, tal que, γ es la razón de aspecto.
$\mathbf{C}_G(u, v)$	Matriz de Segundo Momento, definida como una matriz de tamaño 2x2, que contiene los estimados de la varianza y covarianza de las regiones extraidas de $\mathbf{G}_x$ y $\mathbf{G}_y$ centradas en (u, v), promediadas sobre la máscara Gaussiana $\mathbf{W}_{GS}$.
$\mathbf{\Lambda}_G$	Matriz correspondiente a los eingenvalores $\{\lambda_1, \lambda_2\}$ de $\mathbf{C}_G$.
$\mathbf{\Phi}_G$	Matriz correspondiente a los eingenvectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_2$ de $\mathbf{C}_G$.
$\mathbf{M}_H(u, v)$	Matriz Hessiana, definida como una matriz de tamaño 2x2, que contiene las expresiones discretas de las derivadas parciales de orden superior calculadas sobre el píxel (u, v) de $\mathbf{I}$.
$\mathbf{\Lambda}_H$	Matriz correspondiente a los eingenvalores $\{\lambda_1, \lambda_2\}$de $\mathbf{M}_H$.
$\mathbf{\Phi}_H$	Matriz correspondiente a los eingenvectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_2$ de $\mathbf{M}_H$.

Resumen

El presente trabajo de tesis describe nuestro desarrollo de algoritmos computacionales para el reconocimiento de personas por huellas dactilares, hemos tomado como punto de partida las más consolidadas técnicas y algoritmos publicados por la comunidad científica internacional hasta la fecha, siendo comparables los resultados reportados con los obtenidos.

Además hemos plasmado este trabajo en dos plataformas de hardware:

(a) En un computador personal, implementándose una biblioteca de funciones (toolbox) especializada para el análisis, procesamiento y reconocimiento de personas por huellas dactilares, la que hemos denominado TReHD y además

(b) En una tarjeta de desarrollo DSP, donde se implementó la técnica de verificación de personas basada en un banco de filtros bidimensional.

De los resultados obtenidos podemos inferir que es posible generar tecnología propia en el campo de los sistemas biométricos basados en huellas dactilares, que pueda llegar a ser competitiva y tenga posibilidades de promover el desarrollo de una industria nacional. Por lo que hemos fijado las bases para investigaciones y desarrollos futuros. Esto constituye nuestro principal aporte al desarrollo científico y tecnológico de la Universidad Nacional Mayor de San Marcos y de nuestro país.

Abstract

This work describes our computational algorithm development for fingerprint people recognition. We have taken as a baseline the most strengthen techniques and algorithms published by the International Scientific Community until this time, being comparable the reported results with those obtained.

Additionally, we have implemented this work in two hardware platform:

(a) In a personal computer, in which we have done a Toolbox skilled in analysis, processing and fingerprint people recognition, that we called TReHD.

(b) In a DSP development board, in which we implemented a fingerprint verification technique based on a two-dimensional filterbank.

With the obtained results we can assume that is possible generate own technology in biometric system field based in fingerprint. This technology can become competitive and have possibilities to promote the national industry development. We have set a baseline for future research. This is our principal scientific and technological contribution to the San Marcos University and to our nation.

Capítulo I
Descripción de la Tesis

1.1. Introducción

El presente trabajo describe nuestro desarrollo de algoritmos computacionales para el reconocimiento de personas por huellas dactilares y su implementación en una biblioteca de funciones (*Toolbox*) especializada para el procesamiento y reconocimiento por huellas dactilares, además de la implementación de una técnica de verificación empleando una tarjeta de desarrollo DSP.

1.2. Contexto y motivación

Desde hace algunos años, los sistemas automáticos de identificación biométrica de personas eran exclusivamente utilizados por instituciones forenses y/o gubernamentales como consecuencia de los enormes costos que involucraba esta tecnología, totalmente cerrada a personas e instituciones civiles. Hoy en día, el enorme crecimiento de la tecnología electrónica, las redes de computadoras y las tecnologías de información y comunicaciones (TICs) ha permitido que nuestra sociedad actual esté caracterizada por la comunicación mutua, globalizada e inmediata de personas naturales, entidades gubernamentales, académicas, bancarias, empresariales, entre otras a través de un sistema de interconexión vulnerable. Esto ha favorecido a que el fraude producido por falsa identidad esté alcanzando proporciones nunca antes vistas. Por ello, existe un creciente énfasis en el surgimiento de aplicaciones de identificación automática de personas basadas en sus huellas dactilares, puesto que es una de las alternativas más consolidada y fiable en la actualidad.

El reporte del mercado biométrico del año 2004 realizado por el Grupo Biométrico Internacional (*International Biometric Group* - IBG) revela que los siste-

mas biométricos basados en huellas dactilares continúan siendo la tecnología líder en términos de la proporción del mercado, comandando el 48% de los ingresos biométricos (ver figura 1.1).

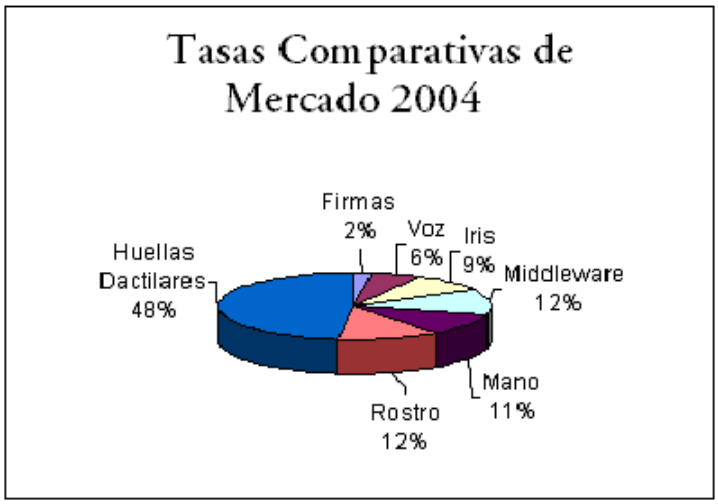


Figura 1.1 Mercado comparativo de los sistemas biométricos [URL 1.1].

Por otro lado la figura 1.2 nos da una idea de las rentas producidas y proyectadas para las tecnologías biométricas para los años 2003 al 2008. Se espera que crezcan de 719 millones de dólares americanos en el año 2003 a 4 mil millones de dólares americanos en el año 2008.

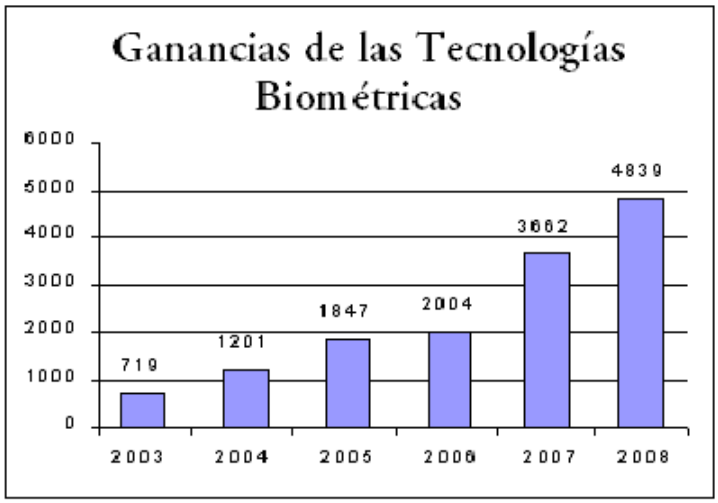


Figura 1.2 Ganancias de las tecnologías biométricas [URL 1.1].

Por lo que se puede concluir que los sistemas biométricos en general, y especialmente los sistemas de reconocimiento automático basado en huellas dactilares, representan una importante área de investigación y desarrollo tecnológico, ya que tiene un amplio campo de aplicación y un mercado potencial creciente en los últimos años.

1.3. Definición del problema

A pesar de que en los países desarrollados se invierten grandes sumas de dinero en investigación a fin de mejorar las técnicas y producir equipos biométricos basados en huellas dactilares a escalas industriales (debido a su alta rentabilidad), en nuestro país la tecnología con la que contamos es importada, evitándose así la posibilidad de crear tecnología propia. Indudablemente no es nada sencillo competir con otros sistemas hechos por países desarrollados, pero este tipo de trabajo, a parte de darnos la ventaja de ofrecer un aporte al desarrollo tecnológico de nuestro país, nos permite formarnos como investigadores especializados y capacitar a otros investigadores.

Además, existe la concepción errónea de que el reconocimiento automático de personas por huellas dactilares es un problema totalmente resuelto, debido a que ha sido una de las primeras aplicaciones de reconocimiento automático desde hace casi cuarenta años. Por el contrario, las metodologías computacionales requeridas para esta tarea siguen siendo en la actualidad un desafiante e importante problema de reconocimiento de patrones [Maltoni, et al., 2003, pp. 48 - 51].

1.4. Objetivos

Este trabajo tiene como objetivo fundamental fijar las bases para el desarrollo de sistemas biométricos basados en huellas dactilares, a través de la implementación de consolidadas técnicas y algoritmos existentes en la actualidad, publicados por la comunidad científica internacional, evaluando su funcionamiento y proponiendo mejoras para incrementar la robustez y eficiencia en cada etapa involucrada del proceso. En el trabajo se identifican dos objetivos específicos:

Desarrollar una biblioteca de funciones (*Toolbox*) que permita analizar,

probar y mejorar algoritmos de procesamiento y reconocimiento de imágenes de huellas dactilares, previo a su implementación para fines de investigación, comerciales u otros.

Implementar una técnica de verificación de personas por sus huellas dactilares en una tarjeta de desarrollo DSP, previamente validada por la herramienta gráfica propuesta.

1.5. Metodología de la Investigación

Para alcanzar los objetivos planteados en el presente trabajo, seguimos la metodología de la investigación expuesta en la figura 1.3. Esta metodología dispone de tres elementos metodológicos. Por una parte, una Introducción Teórica que permite establecer los conceptos fundamentales vinculados a los sistemas biométricos basados en huellas dactilares. Por otra parte, el Desarrollo Algorítmico y la Implementación son elementos metodológicos vinculados, que permiten integrar las etapas de planteamiento, elaboración, optimización y validación de los algoritmos con su implementación en dos plataformas de hardware específicas.

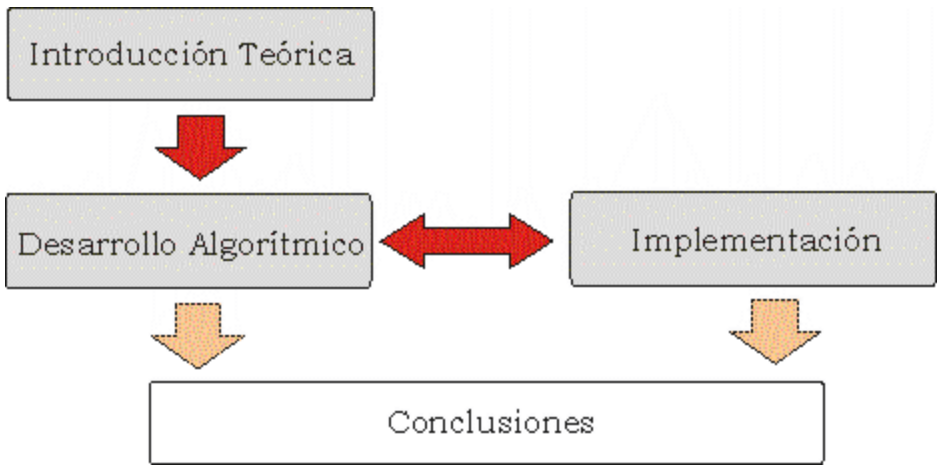


Figura 1.3 Esquema general de la metodología de investigación planteada en este trabajo.

La Introducción Teórica se divide en dos etapas (ver figura 1.4). En la primera etapa se describen los principales fundamentos de los sistemas biométricos en

general y los basados en huellas dactilares en particular, incluyendo valiosos conceptos y definiciones, además de una estricta, rigurosa y completa terminología necesaria para comprender este trabajo. En la segunda etapa se describen aquellas técnicas algorítmicas empleadas vinculadas al procesamiento de imágenes y al reconocimiento de patrones, es importante recalcar que la implementación de un sistema biométrico requiere un profundo conocimiento de ambas ramas de la ciencia computacional.

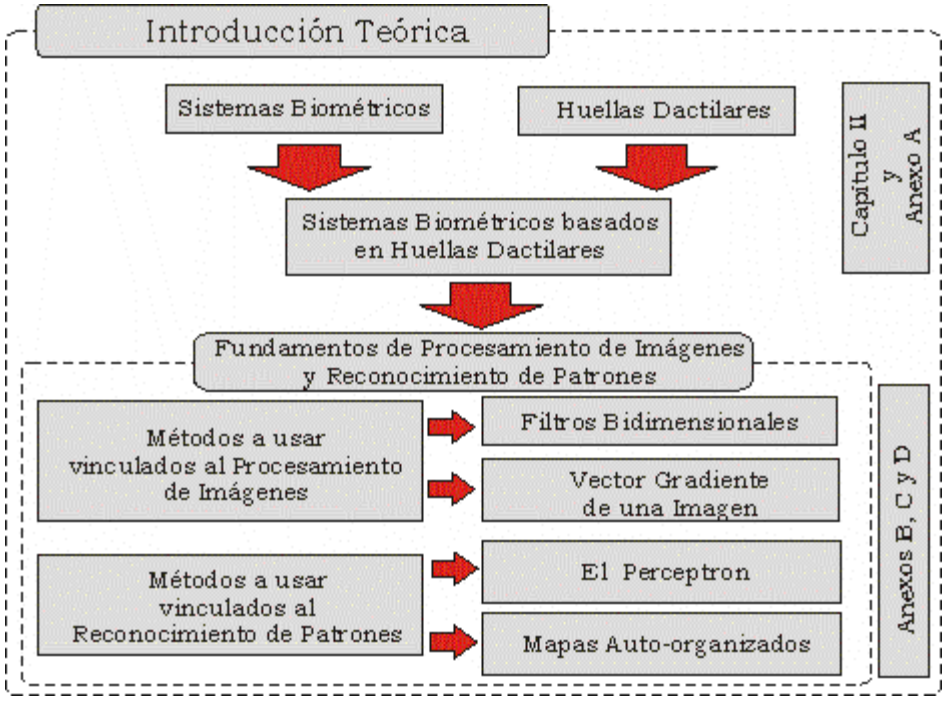


Figura 1.4 Esquema detallado de la Introducción Teórica.

El Desarrollo Algorítmico se realizó a partir de una amplia revisión de la literatura especializada que recoge los trabajos de investigación relacionados. Hemos establecido que la elaboración algorítmica de un sistema biométrico basado en huellas dactilares requiere de tres etapas generales: Análisis, Extracción de Caracteristicas y Coincidencia de impresiones dactilares (ver figura 1.5), tal sub división queda sustentada en la estructura general de todo sistema de reconocimiento de patrones. Una etapa de Análisis, que permite la descomposición de la estructura intrínseca del patrón de entrada en nuevos patrones que particularizan algún tipo de información presente. La etapa de Extracción de Características, que permite representar al patrón en forma sintetizada valiéndose de las salidas gene-

radas por la etapa de Análisis y por el patrón mismo. Por último, la etapa de Coincidencia, permite establecer la semejanza del patrón de entrada con algún otro patrón previamente registrado, valiéndose de las salidas generadas por la etapa de Extracción de Características.

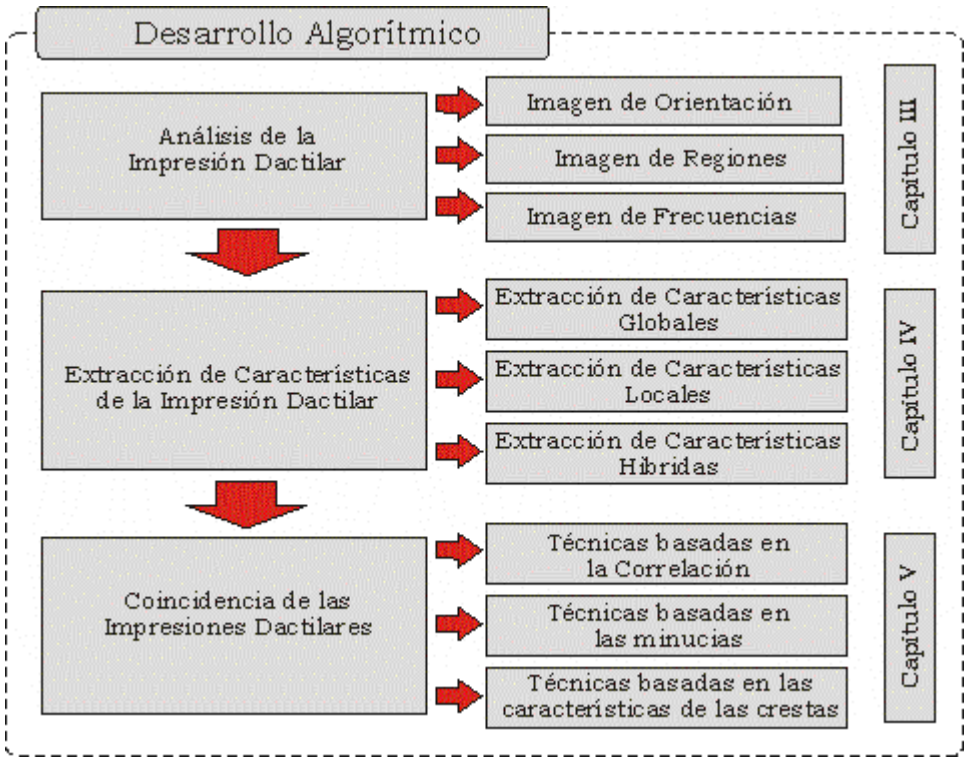


Figura 1.5 Esquema detallado del Desarrollo Algorítmico.

La Implementacion, elemento metodológico vinculado al Desarrollo algorítmico permite plasmar en dos plataformas de hardware las diferentes técnicas desarrolladas en el presente trabajo (ver figura 1.6), además permite la validación y evaluación del desempeño de cada técnica desarrollada, empleando para ello una base de datos estándar (ver sección 5.5.1) y las medidas estándares de rendimiento establecidas por la comunidad científica (ver sección 2.5).

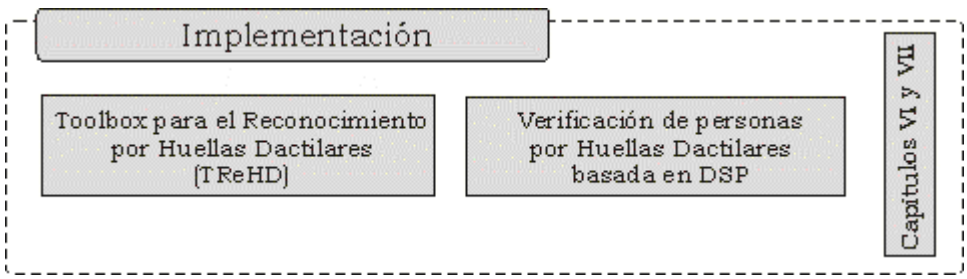


Figura 1.6 Esquema detallado de la Implementación.

Por último, se plantearon las conclusiones del trabajo y propuestas para futuros trabajos de investigación.

1.6. Alcance de la investigación

Este trabajo se restringe únicamente al desarrollo de algoritmos para el reconocimiento de personas por huellas dactilares, pues representan la parte fundamental para el diseño de todo sistema automático de reconocimiento.

1.7. Aportes

Establecer en la Facultad de Ingeniería Electrónica las bases para el desarrollo de herramientas computacionales que agilicen el diseño de sistemas biométricos basados en huellas dactilares, así como también para otras técnicas de reconocimiento apoyadas en imágenes.

Fijar las bases en la construcción de un prototipo altamente fiable para la verificación de huellas dactilares, competitivo y accesible en el mercado nacional que tenga posibilidades de promover el desarrollo de una industria nacional en el campo de la biométrica basada en huellas dactilares.

Aplicar sofisticadas tecnologías que involucran áreas tales como la visión por computador, el reconocimiento estadístico de patrones y la inteligencia computacional. Este conocimiento basado en la experiencia de un problema complejo real, nos ha valido el reconocimiento de otras instituciones, tales como por ejemplo el Instituto Peruano de Energía Nuclear y la Universidad Nacional de Ingeniería quienes nos eligieron para formar parte en el desarrollo de proyectos de investigación lo que nos ha permitido resolver otras importantes tareas de análisis de imágenes y reconocimiento de patrones.

Realizar ponencias y generar publicaciones con el objetivo de difundir el nivel de investigación y desarrollo de la Facultad de Ingeniería Electrónica. Como por ejemplo se ha tenido la oportunidad de exponer en el Congreso INTERCON 2003 y en el Seminario Taller de Investigación FIE-UNMSM 2003 el tema del reco-

nocimiento biométrico por huellas dactilares utilizando la representación basada en un banco de filtros bidimensional, así como también se han publicado en la revista Electrónica de nuestra facultad Nº 12 y 13, dos artículos acerca de los algoritmos para el reconocimiento y la segmentación de imágenes de huellas dactilares, quedando aún pendiente próximas publicaciones y presentaciones que serán presentadas a nombre de la universidad y la facultad.

1.8. Componente económico

La elaboración del presente trabajo ha significado destinar recursos para la adquisición de materiales de software, hardware y otros, cuyos costos están expresados en la tabla 1.1.

Hardware	Costo (US$)
Computador Pentium MMX	200.00
Computador Pentium IV	950.00
DSP Starter Kit TMS320C6711	380.00
Software	
Windows 98 SE	40.00
Visual Studio 6.0	30.00
Matlab 6.5	50.00
Base de datos FVC 2000 y 2002	100.00
Soporte	
Material Bibliográfico	500.00
Conexión a Internet	1440.00

Tabla 1.1 Costos de materiales empleados.

La tabla anterior no incluye costos de desarrollo, como son horas/hombre en investigación y programación puesto que para el caso, omitimos la valoración económica de este rubro.

1.9. Organización de la tesis

Este trabajo está organizado de la siguiente manera:

El capítulo II presenta una descripción general de los sistemas biométricos especificando su arquitectura, sus modos de operación, sus medidas estándares de rendimiento y las tecnologías biométricas en las que se basan;

en especial se detallan las características de los sistemas basados en huellas dactilares.

El capítulo III describe el cálculo de la Imagen de Orientación, la Imagen de Frecuencias y la Imagen de Regiones de una impresión dactilar, pues cada una de éstas resume cierta información particular de la huella.

El capítulo IV describe las principales técnicas para la extracción de características como son los puntos singulares, las minucias y un patrón netamente estadístico denominado *Finger Code*.

El capítulo V está dedicado a exponer tres técnicas para la coincidencias de impresiones dactilares basadas en la correlación, alineamiento de puntos y distancia Euclidiana, estableciéndose una comparación de su rendimiento.

El capítulo VI presenta y describe la biblioteca de funciones especializada para el reconocimiento de personas por huellas dactilares, denominada TReHD.

El capítulo VII describe la implementación de una técnica de verificación de personas por huellas dactilares a través del procesador DSP de alto rendimiento TMS320C6711.

En el capítulo VIII se presentan las conclusiones de este trabajo junto con las propuestas de trabajos futuros.

En las siguientes secciones de la tesis se incluyen cinco anexos necesarios para la completa comprensión del texto, tratando temas como, captura de huellas dactilares, filtros bidimensionales, Vector Gradiente, redes neurales y el DSP *Starter Kit* TMS320C6711.

Capítulo II
Sistemas Biométricos y Huellas Dactilares

2.1. Introducción

Tradicionalmente en aquellas aplicaciones que requieren un cierto grado de seguridad se utilizan dos métodos para el reconocimiento automático de personas: por posesión y por conocimiento. En el Método por Posesión, la persona dispone de un objeto que le permite identificarse, tal como por ejemplo: una licencia de conducir, un documento de identidad, una tarjeta de crédito, entre otros. En el Método por Conocimiento, la persona es identificada por una clave o número de identificación personal (*Personal Identification Number* – PIN). Ambos métodos presentan múltiples inconvenientes debido a que no son capaces de diferenciar adecuadamente entre una persona autorizada y una impostora, por ejemplo las tarjetas de identificación pueden ser robadas o extraviadas y las claves pueden ser olvidadas o adivinadas. Por otra parte, existe el Método Biométrico que permite utilizar las informaciones intrínsecas a la persona que son difícilmente transferibles o extraídas, conocidas como características biométricas. Por consiguiente este método presenta niveles de confianza más altos que los tradicionales, puesto que utiliza características únicas que requieren la presencia física de la persona a identificar.

2.2. Sistemas biométricos

El reconocimiento biométrico se refiere al uso de una característica distintiva al individuo que permita reconocerlo automáticamente, dicha característica es usualmente denominada como identificador biométrico o simplemente "biométrica". Un sistema biométrico es básicamente un sistema de reconocimiento de patrones que reconoce a un individuo estableciendo la autenticidad de la biométrica que posee. Éste opera adquiriendo los datos biométricos del individuo, extrayendo la

plantilla de entrada y comparándola con la(s) plantilla(s) previamente registrada(s) en la base de datos del sistema.

El término plantilla es empleado en este trabajo haciendo referencia únicamente al modelo representativo del identificador biométrico, sin distinguir si ésta es capturada en línea o la almacenada en la base de datos, debido a que la traducción de *template* posee múltiples significados. Como una plantilla está constituida por un subconjunto de la información contenida en la biométrica, no es posible reconstruir la biométrica a partir de la plantilla, manteniéndose así un alto nivel de seguridad.

2.2.1. Arquitectura

Un sistema biométrico se diseña utilizando como base los siguientes cinco módulos (ver figura 2.1):

Módulo de captura: Permite adquirir el dato biométrico de un individuo.

Módulo de extracción de características: El dato adquirido es procesado para extraer la plantilla de entrada o conjunto de características discriminatorias.

Módulo de coincidencia: La plantilla de entrada es comparada con la(s) plantilla(s) almacenada(s), generando una puntuación sobre la comparación.

Módulo de base de datos: Es usado para almacenar las plantillas de los usuarios registrados o inscritos en el sistema biométrico. Dependiendo de la aplicación la plantilla puede ser almacenada en una base de datos central o registrada en una *smart card* emitida para el individuo. Usualmente son almacenadas múltiples plantillas de un individuo para tomar en cuenta las variaciones en la biométrica, donde además éstas pueden ser actualizadas en el tiempo.

Módulo de toma de decisiones: La identidad del individuo es declarada o aceptada/rechazada en base a la puntuación de la comparación o comparaciones.

Estos módulos en conjunto realizan dos tareas principales:

Tarea de Inscripción: El sistema registra a un nuevo usuario autorizado por el administrador del sistema, almacenando en la base de datos la plantilla de entrada y registrando la identidad del nuevo usuario.

Tarea de Reconocimiento: El sistema toma una decisión acerca de la certeza de la identidad de un individuo comparando la plantilla de entrada con la(s) previamente almacenada(s) en la base de datos.

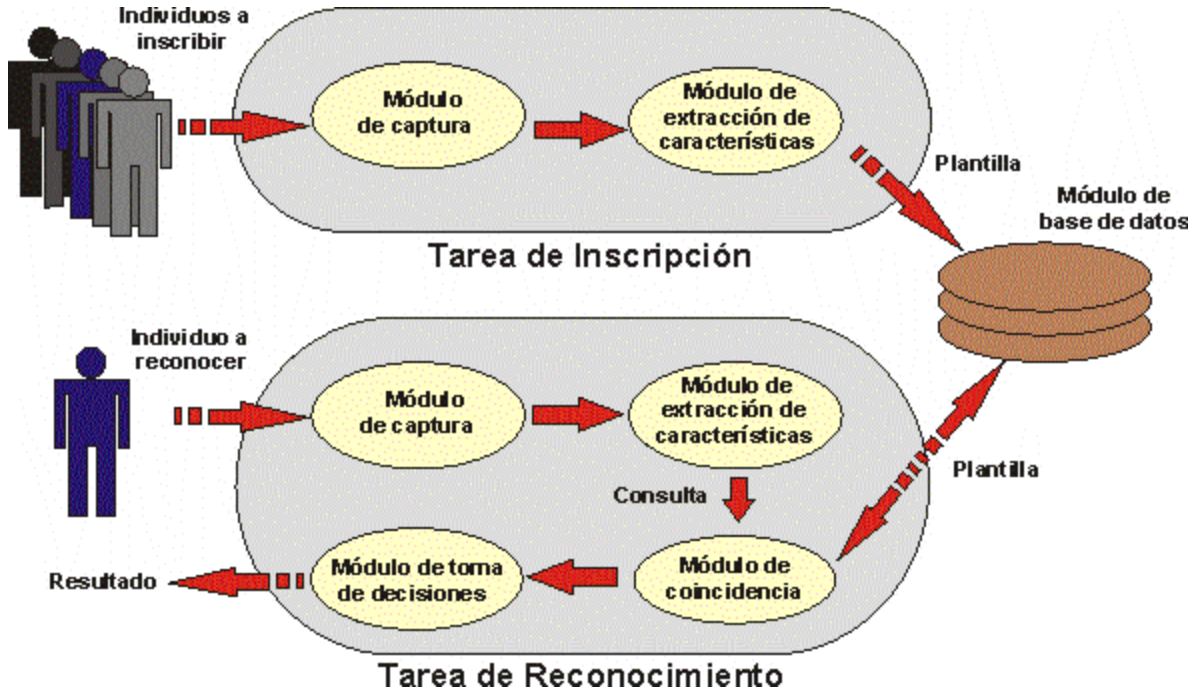


Figura 2.1 Arquitectura de un Sistema de Reconocimiento Biométrico.

2.2.2. Modos de operación

Dependiendo del contexto de la aplicación, la tarea de reconocimiento puede trabajar en los siguientes modos:

Modo de verificación. El sistema valida la identidad de un individuo comparando la plantilla de entrada con su plantilla correspondiente previamente almacenada en la base de datos. En este caso el individuo que desea ser reconocido declara una identidad al sistema, usualmente a través de un PIN, un nombre de usuario o una *smart card* y luego se realiza una comparación uno a uno para determinar si la identidad declarada es

verdadera o no. La verificación de la identidad es típicamente usada para el reconocimiento positivo, en donde el objetivo es impedir que múltiples personas usen la misma identidad.

Modo de identificación. El sistema identifica a un individuo comparando la plantilla de entrada con las plantillas de todos los usuarios registrados en la base de datos, es decir se realiza una comparación uno a muchos para establecer la identidad del individuo sin que ésta sea declarada. La identificación es un componente crítico en aplicaciones de reconocimiento negativo en donde el sistema establece si la persona es quien explícita o implícitamente niega ser. El propósito del reconocimiento negativo es impedir que una sola persona use múltiples identidades. La identificación también puede ser usada para el reconocimiento positivo en donde el usuario no requiere declarar una identidad.

El término autenticación es utilizado usualmente como sinónimo de verificación, sin embargo en el lenguaje de las tecnologías de información, autenticar significa hacer saber al sistema la identidad del usuario sin importar el modo de operación. A lo largo de este trabajo usaremos el término genérico reconocimiento para referirnos indistintamente al proceso de identificación o verificación. Los diagramas de bloques de un sistema de verificación e identificación son mostrados en las figuras 2.2 y 2.3.

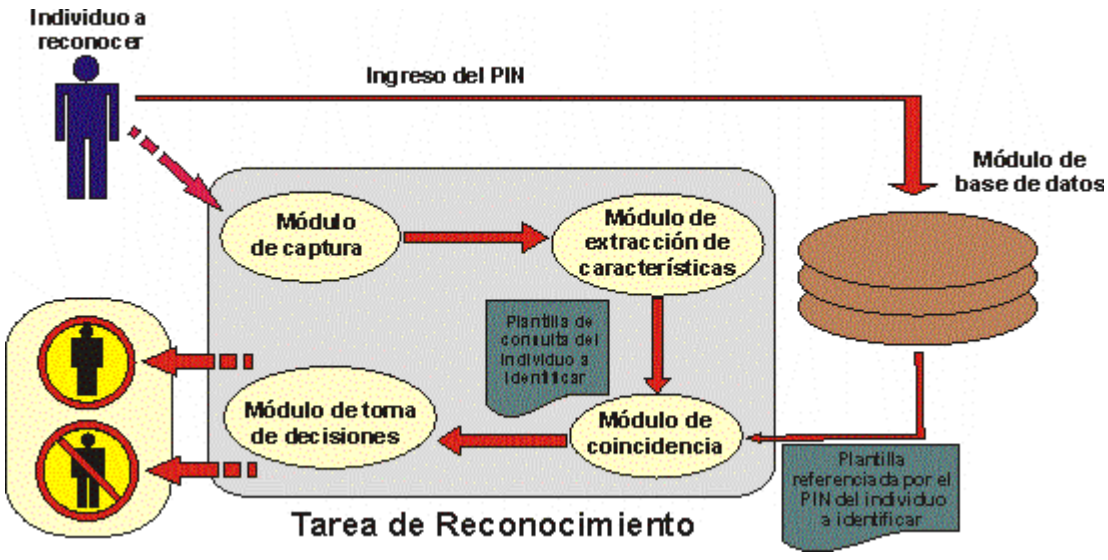


Figura 2.2 Tarea de reconocimiento en el modo verificación.

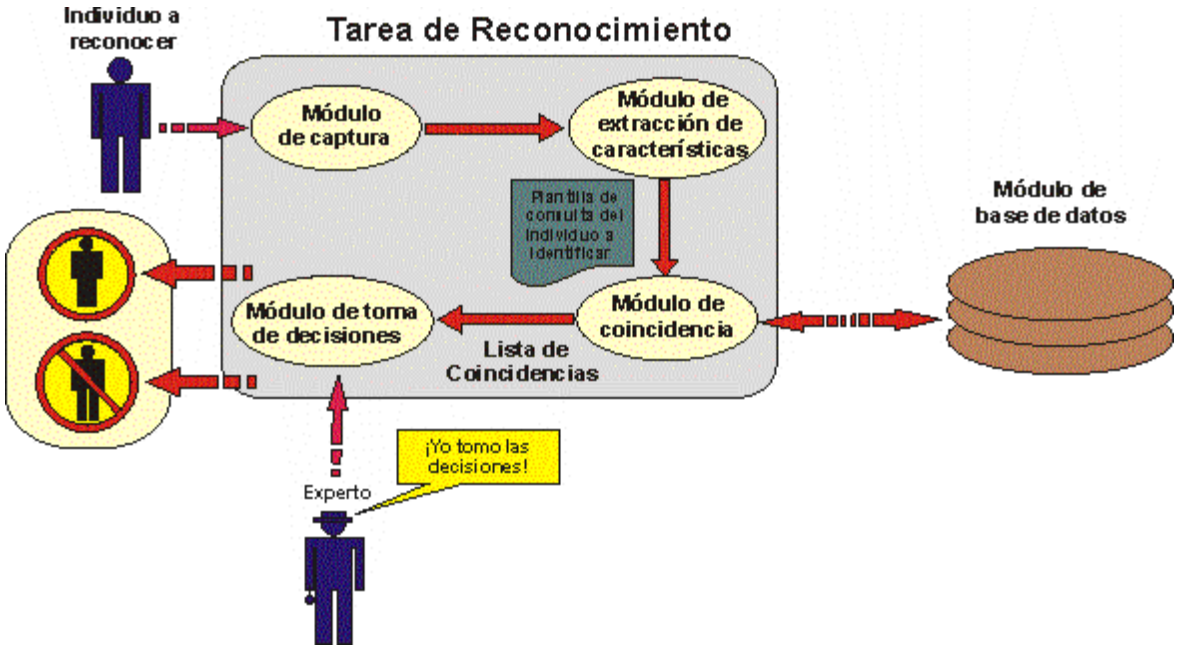


Figura 2.3 Tarea de reconocimiento en el modo identificación.

2.2.3. Tecnologías biométricas

La medida biológica usada como identificador biométrico es cualquier característica fisiológica o psicológica que cumpla con las propiedades de ser universal, distintiva, permanente y medible cuantitativamente. Existe una gran variedad de identificadores biométricos usados en muchas aplicaciones (ver figura 2.4). Cada biométrica tiene sus fortalezas y debilidades, la elección depende típicamente de las características de la aplicación y de las propiedades de la biométrica. Las biométricas comúnmente utilizadas son: el rostro, el termograma del rostro, las huellas dactilares, la geometría de la mano, los patrones del iris, retina, voz y la firma. Entre todos los identificadores biométricos, las huellas dactilares poseen los niveles más altos de confiabilidad, pues es una de las tecnologías biométricas más maduras y adecuada para un gran número de aplicaciones, tal como se refleja en el reporte de las rentas generadas por varias tecnologías biométricas en el año 2004 realizado por el IBG (ver Figura 1.2.a), además en la actualidad existe un creciente uso en aplicaciones civiles y de alta seguridad.

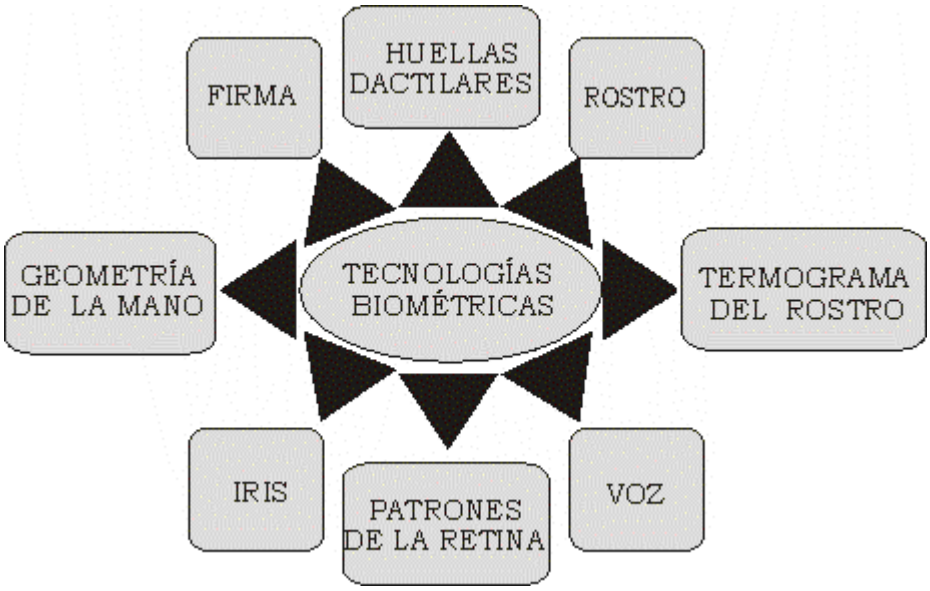


Figura 2.4 Principales tendencias en el desarrollo de las tecnologías biométricas.

En la práctica para el diseño de un sistema biométrico existen importantes consideraciones, tales como el desempeño (precisión, velocidad y robustez), aceptación y poca vulnerabilidad ante fraudes y ataques. Además, es necesario considerar también aspectos tales como el costo de los dispositivos de adquisición, la probabilidad de error, la facilidad de uso por el usuario, entre otros.

2.3. Huellas dactilares

Las huellas dactilares son patrones constituidos por las crestas papilares de los dedos de las manos, se localizan en la dermis y se reproducen en la epidermis (ver figura 2.5.a), generando configuraciones diversas. Se forman en el período fetal, a partir del sexto mes, manteniéndose invariables a través de la vida del individuo, a menos que sufran alteraciones debido a accidentes tales como cortes o quemaduras. Las impresiones dactilares son las reproducciones resultantes de las huellas sobre una superficie plana, quedando almacenada en formato analógico (papel) o digital (archivo), en éstas las crestas papilares se aprecian como las líneas más oscuras y los surcos o valles interpapilares como las líneas más claras (ver figura 2.5.b).

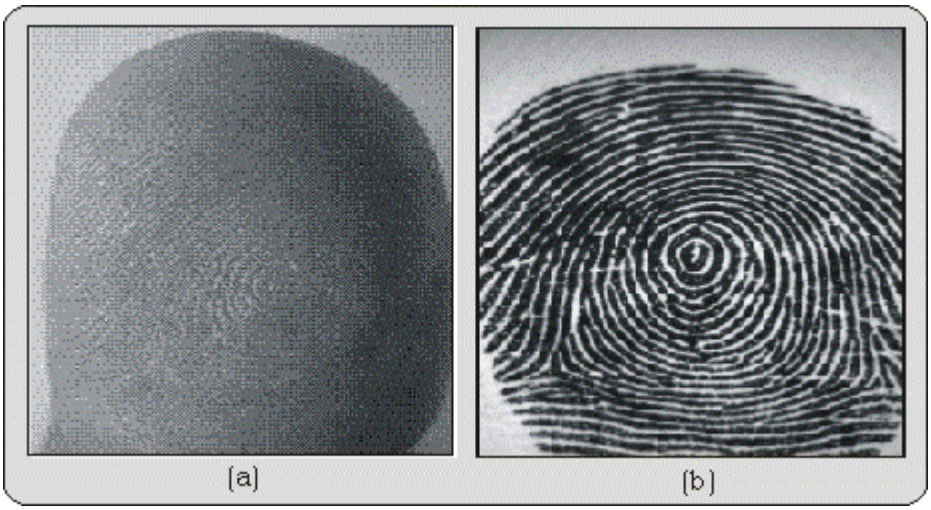


Figura 2.5 (a) Huella dactilar, figura 1 de [Khuwaja y Abu-Rezq, 2003]; (b) impresión dactilar, imagen 18_6.tif de la base de datos 2 del FVC2002.

2.3.1. Características

Los expertos han definido dos tipos de características, comúnmente denominadas globales y locales. Veamos cada una de éstas.

2.3.1.1. Características globales

Son los tipos de patrones geométricos de las crestas que son reconocibles a simple vista, basados en estos patrones las huellas dactilares se dividen en tres grupos principales: *Loop*, *Arch* y *Whorl*. Estos grupos también se subdividen en grupos más pequeños, como el caso del *Loop* que se subdivide en *Left loop* y *Right loop*. Usualmente la determinación del patrón al que pertenece la huella dactilar se obtiene mediante el conocimiento de sus puntos singulares. Es necesario por lo tanto explicar dos conceptos relacionados.

Área Patrón: Es la parte principal de la huella dactilar y está constituida por las crestas y todas sus características.

Líneas Tipo: Son definidas como dos crestas que se inician paralelamente y divergen sobre el área patrón. Estas crestas pueden ser continuas o no, en caso de que ocurra alguna ruptura (ver figura 2.6).

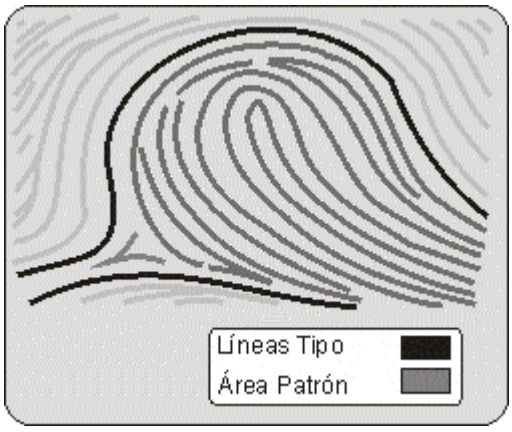


Figura 2.6 Área Patrón y Líneas Tipo.

2.3.1.1.1. Puntos singulares

Punto *Core*: Está localizado dentro del Área Patrón en donde las crestas presentan una mayor curvatura. Debido a la gran variación en las configuraciones de las crestas (ver figura 2.7.a), las técnicas para su determinación automática son muy complejas.

Punto *Delta*: Es el punto de divergencia de las Líneas Tipo más internas que tienden a envolver el Área Patrón. Un *Delta* es un triángulo constituido por las crestas papilares que pueden formarse de dos maneras: por la bifurcación de una línea simple o por la brusca divergencia de dos líneas paralelas (ver figura 2.7.b).

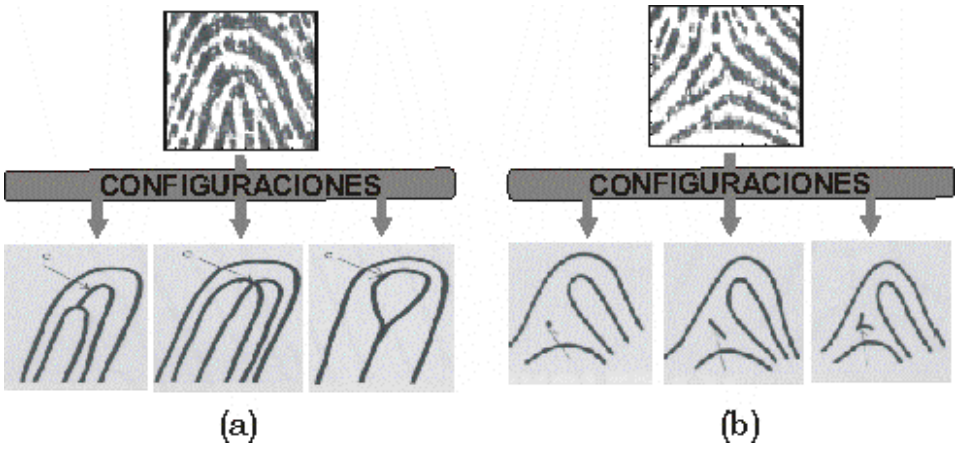


Figura 2.7 Puntos singulares (a) configuraciones del punto *Core*; (b) configuraciones del punto *Delta*.

2.3.1.2. Características locales

Las características locales establecen la individualidad de la huella dactilar y están representadas por los puntos conocidos como minucias. Es posible que dos o más individuos tengan huellas dactilares con idénticas características globales, pero seguirán siendo distintas y únicas debido a que poseen diferentes características locales, es decir éstos son elementos distintivos que caracterizan a una huella dactilar como un objeto único.

2.3.1.2.1. Minucias

Las crestas en una huella dactilar no son continuas ni rectas, sino más bien cambian de dirección, cortándose y bifurcándose. Los puntos en donde los cambios ocurren son denominados minucias. En una imagen de alta calidad es común encontrar entre setenta y cien minucias, las cuales proveen la suficiente información para determinar la individualidad de una huella dactilar. Los tipos de minucias son (ver figura 2.8):

1. **Laguna** (*Enclosure*): Es una cresta que se divide en dos ramas y se unifica otra vez luego de recorrer una distancia corta creando un área cerrada.
2. **Isla** (*Dot*). Es una cresta muy pequeña, a tal grado que es semejante a un punto.
3. **Cresta Independiente** (*Short ridge*): Es una cresta muy corta pero lo suficientemente grande para no ser una isla.
4. **Aguijón** (*Spur*): Es una cresta que se divide en dos ramas y una de éstas recorre una distancia muy corta finalizando.
5. **Finalización** (*Ending*): Es el punto en donde una cresta termina abruptamente.
6. **Bifurcación** (*Bifurcation*): Es el punto en donde una cresta se divide en dos ramas.
7. **Trifurcación** (*Crossover*): Es producida por la unión de dos minucias de bifurcación.

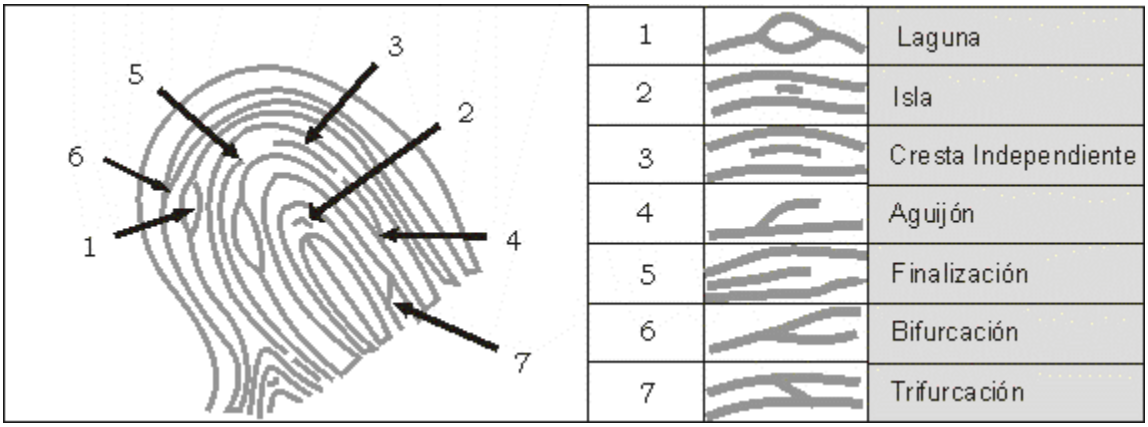


Figura 2.8 Tipos de minucias.

Las minucias son clasificadas en dos categorías: tipos básicos y compuestos. Los tipos compuestos están constituidos a partir de los tipos básicos. Las minucias de finalización y bifurcación son consideradas como los tipos básicos y el resto de las minucias son consideradas como los tipos compuestos. Por ejemplo, una laguna es la unión de dos bifurcaciones de direcciones opuestas; una cresta corta es la unión de dos minucias de finalización con una separación muy pequeña.

Los principales atributos de una minucia son:

Dirección: Cada minucia posee una dirección particular, siendo ésta concordante con la dirección de la cresta a la que pertenece.

Posición: Es la ubicación geométrica de la minucia con respecto al eje de referencia de la imagen o con respecto a algún punto de referencia.

Frecuencia espacial: Es la inversa de la distancia entre dos crestas consecutivas en la vecindad de una minucia.

Curvatura: Es el índice de variación en la dirección de las crestas.

El tipo, posición y dirección de cada minucia son almacenadas para la tarea de reconocimiento. La figura 2.9 muestra la dirección y posición de las minucias de bifurcación y finalización respecto a los ejes de referencia de la imagen.

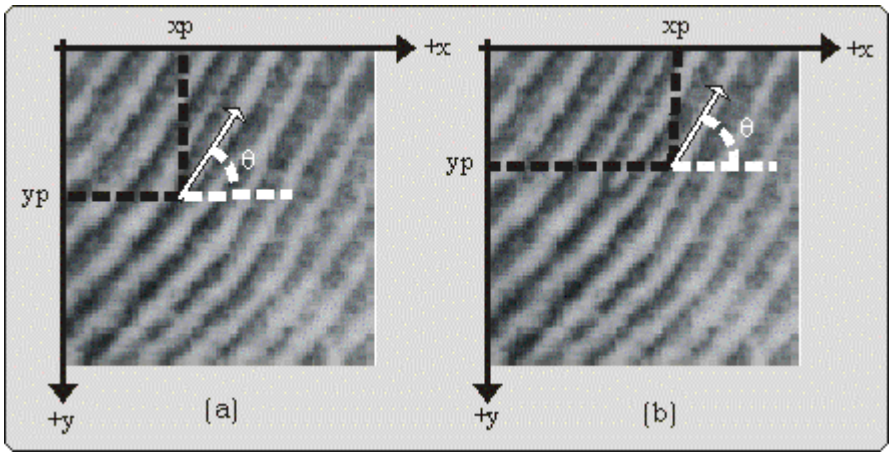


Figura 2.9 Posición y dirección de las minucias básicas (a) minucia de bifurcación; (b) minucia de finalización.

2.4. Sistemas biométricos basados en las huellas dactilares

Son sistemas que fundamentan sus decisiones de reconocimiento tomando como característica personal a la huella dactilar. De acuerdo con el modo de operación en que trabajen, éstos son conocidos en la literatura como:

Sistema Automático de Identificación por Huellas Dactilares (*Automatic Fingerprint Identification System* - AFIS)

Sistema Automático de Verificación por Huellas Dactilares (*Automatic Fingerprint Authentication System* - AFAS)

Tradicionalmente la verificación e identificación por huellas dactilares es realizada manualmente por un experto, sin embargo es una tarea tediosa, lenta y costosa que no presenta los requerimientos de desempeño necesarios para las aplicaciones actuales. Como consecuencia los AFIS tienen una gran demanda en el mercado, sin embargo la tecnología aplicada a estos sistemas sigue estando aún en desarrollo pues todavía existen algunos problemas por resolver.

Las principales etapas consideradas en el diseño de un AFIS son:

Captura y almacenamiento. Siendo un AFIS básicamente un sistema que trabaja con impresiones dactilares es necesario definir los principales parámetros que las caracterizan, tales como: la resolución, el área, el número

de píxeles por pulgada, contraste y distorsión geométrica. También puede considerarse aquí la técnica de compresión a utilizar para el almacenamiento, el formato de archivo, entre otros (ver anexo A).

Procesamiento y Representación: Debido al ruido comúnmente presente en las imágenes, producido generalmente por la técnica de adquisición, las características del dispositivo de captura y defectos en la imagen, (ver anexo A) usualmente es necesario utilizar alguna técnica de procesamiento para el mejoramiento o reconstrucción de la imagen. El objetivo de la representación es determinar un espacio de medidas (características) en el que las imágenes pertenecientes al mismo dedo formen un agrupamiento compacto y las que pertenecen a diferentes dedos ocupen diferentes porciones de este espacio. Una buena representación debería contener información distintiva acerca de la impresión dactilar, además de ser fácilmente extraíble y almacenada de forma compacta para la coincidencia. El proceso de obtener una representación se denomina extracción de características y se realiza en varios niveles posibles: globales y locales.

Coincidencia. Consiste en la comparación biunívoca entre dos impresiones dactilares devolviendo su grado de similitud. Es una tarea extremadamente difícil, debido principalmente a la gran variabilidad entre diferentes impresiones de la misma huella producida por: el desplazamiento, rotación, traslape parcial, distorsiones no lineales, presión, condiciones de la piel, el ruido y los errores en la extracción de características. Además las impresiones de diferentes huellas pueden parecer similares, especialmente en términos de su estructura global.

Clasificación e indexado. La gran cantidad de huellas dactilares captadas y almacenadas en un amplio rango de aplicaciones hace poco práctica la comparación exhaustiva de una característica desconocida con todas las registradas en la base de datos, para reducir el tiempo de búsqueda y la complejidad computacional es deseable clasificar las impresiones capturadas de forma consistente y precisa, de tal manera que la comparación se realice sólo con un subconjunto de las plantillas en la base de datos. La clasificación es una técnica utilizada para catalogar a una huella dactilar dentro de una

de las clases establecidas en la literatura. La clasificación también puede considerarse como una coincidencia de las características a nivel global.

El desarrollo de un sistema biométrico basado en huellas dactilares está íntimamente relacionado con el procesamiento digital de imágenes y la teoría del reconocimiento de patrones. Por procesamiento digital de imágenes se entiende la manipulación de una imagen de entrada, de modo que la salida del proceso sea una nueva imagen. Análogamente, para el reconocimiento de patrones, se presenta a la entrada del proceso un patrón (imagen) obteniéndose como salida una categoría o clase. En el caso de huellas dactilares el objetivo final es la comparación de la plantilla vinculada a la impresión dactilar a ser reconocida con la(s) almacenada(s) en la base de datos. Para comprender la mayor parte de los temas tratados en el presente trabajo es necesario conocer los fundamentos de las técnicas de procesamiento de imágenes y reconocimiento de patrones, por esta razón hemos incluido los anexos B, C y D. que resumen algunos de estos importantes temas.

2.5. Errores en sistemas biométricos

La salida resultante del módulo de coincidencia en un sistema de reconocimiento basado en huellas dactilares, típicamente es una puntuación que cuantifica el grado de similitud entre la plantilla de entrada y la plantilla almacenada en la base de datos y puede ser considerada sin pérdida de generalidad dentro del rango de cero a uno. Una puntuación cercana a uno/cero indica que hay una alta/baja certeza de que las dos impresiones provengan de la misma huella. La decisión del sistema es regulada por un valor umbral t. Los pares que generan puntuaciones mayores o iguales que t, se deduce que son "pares de coincidencia" (es decir, que pertenecen a la misma huella), de los pares que generan puntuaciones menores que t se deduce que son "pares de no coincidencia" (es decir, que pertenecen a diferentes huellas).

Un típico sistema biométrico de verificación comete dos tipos de errores: equivocarse al tener la certeza que las impresiones de dos diferentes huellas pro-

vengan del mismo dedo (llamada falsa coincidencia) y al tener la certeza de que dos impresiones de la misma huella provengan de diferentes dedos (llamada falsa no coincidencia). Estos dos tipos de errores son usualmente denotados como falsa aceptación y falso rechazo, sin embargo es necesario hacer una distinción de acuerdo a la aplicación. En sistemas de reconocimiento positivo (por ejemplo, un sistema de control de acceso) una falsa coincidencia determina la falsa aceptación de un impostor, mientras que una falsa no coincidencia causa el falso rechazo de un usuario genuino. Por otro lado, en una aplicación de reconocimiento negativo (por ejemplo, impidiéndoles a los usuarios obtener los beneficios de bienestar bajo falsas identidades) una falsa coincidencia resulta en el rechazo de una petición genuina, mientras que una falsa no coincidencia resulta en la falsa aceptación del intento de un impostor. En consecuencia, la utilización de la notación de "falsa coincidencia/no coincidencia" no depende de la aplicación y además es preferida antes que "falsa aceptación/falso rechazo". Sin embargo, la Tasa de Falsa Aceptación (*False Acceptance Rate* - FAR) y la Tasa de Falso Rechazo (*False Rejection Rate* - FRR) son indicadores ampliamente utilizados en entornos comerciales u otros.

2.5.1. En sistemas de verificación

Desde la perspectiva de diseño, el problema de la verificación biométrica puede ser formulado de la siguiente manera: Sea $\mathbf{T}_S$ la plantilla de la impresión dactilar almacenada en la base de datos (también conocida como impresión secundaria) y $\mathbf{T}_P$ la plantilla de la impresión dactilar adquirida para la verificación (conocida como impresión dactilar primaria). Entonces las hipótesis que pueden ser definidas son:

$H_0 : \mathbf{T}_P \neq \mathbf{T}_S$: La impresión dactilar primaria no es la misma que la impresión dactilar secundaria.

$H_1 : \mathbf{T}_P = \mathbf{T}_S$: La impresión dactilar primaria es la misma que la impresión dactilar secundaria.

Las decisiones asociadas con cada una de las hipótesis son:

D_0 : El individuo es un impostor.

D_1 : El individuo es quien dice ser.

La verificación involucra la coincidencia de $\mathbf{T}_P$ y $\mathbf{T}_S$ usando medidas de similitud. Si los índices de similitud o coincidencia son menores a algún valor umbral t, entonces decidir por D_0, en caso contrario decidir por D_1. La anterior terminología es tomada de la teoría de comunicaciones donde deseamos detectar un mensaje ante la presencia del ruido. H_0 es la hipótesis de que la señal recibida sea ruido y H_1 es la hipótesis de que la señal recibida sea el mensaje mas ruido. Tal formulación para la comprobación de las hipótesis contiene implícitamente dos tipos de errores (ver figura 2.10):

1. Falsa coincidencia (se decide por D_1 cuando se cumple la hipótesis H_0).
2. Falsa no coincidencia (se decide por D_0 cuando se cumple la hipótesis H_1).

La Tasa de Falsa Coincidencia (*False Match Rate* - FMR) es la probabilidad del primer tipo de error y la Tasa de Falsa No Coincidencia (*False Non-Match Rate* - FNMR) es la probabilidad del segundo tipo de error.

Para evaluar la precisión de un sistema biométrico se deben recolectar las puntuaciones generadas por varios pares de impresiones dactilares de la misma huella, generando una distribución conocida como Distribución Genuina. También se deben recolectar las puntuaciones generadas por varios pares de impresiones dactilares de diferentes huellas, tradicionalmente conocida como Distribución Impostora (ver figura 2.11).

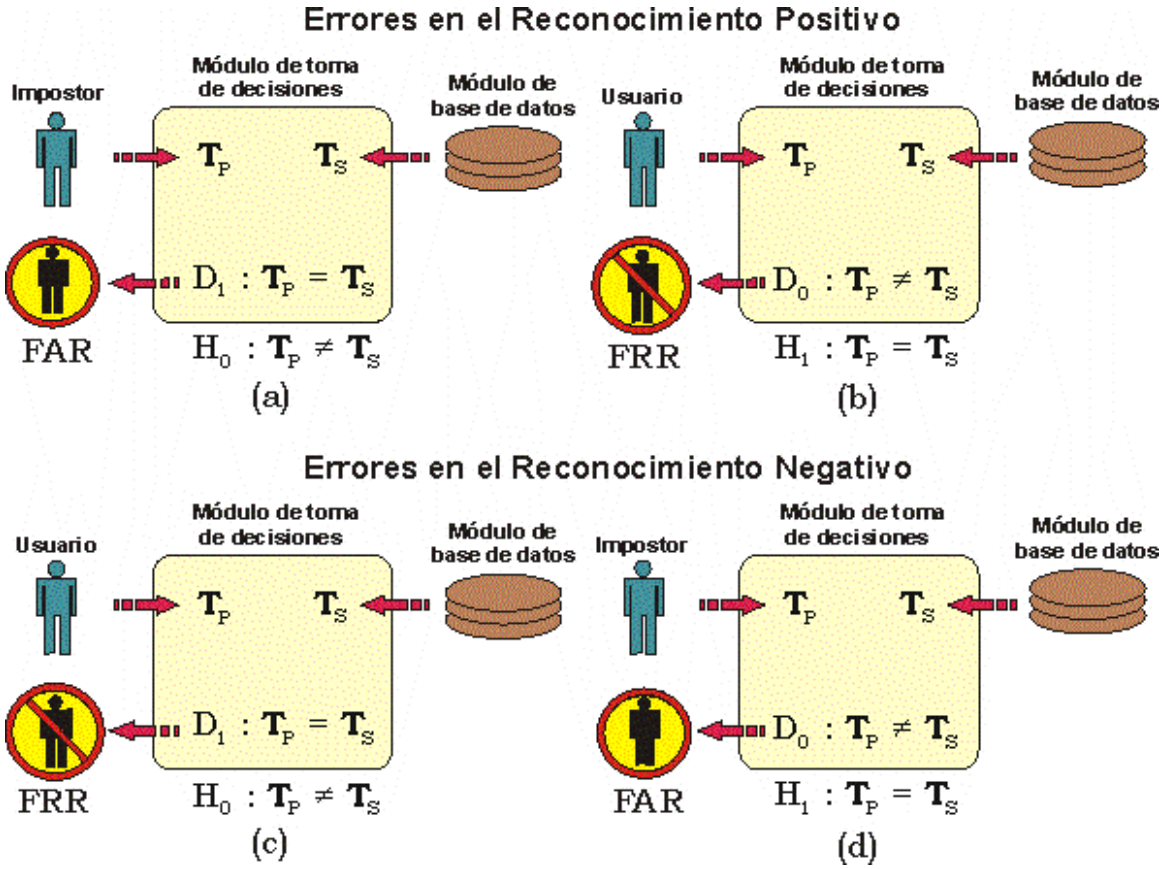


Figura 2.10 Tipos de errores en el reconocimiento (a) falsa coincidencia para reconocimiento positivo; (b) falsa no coincidencia para reconocimiento positivo; (c) falsa coincidencia para reconocimiento negativo; (d) falsa no coincidencia para reconocimiento negativo.

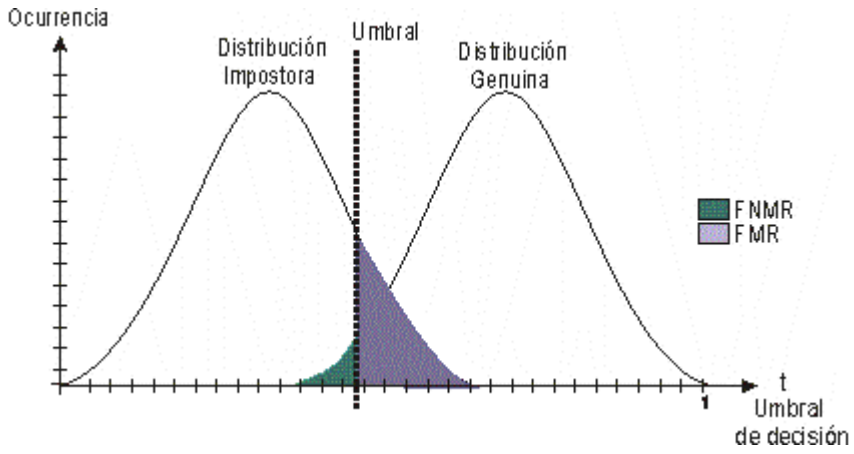


Figura 2.11 Indicadores FMR y FNMR para un umbral de decisión dado, mostrados sobre las Distribuciones Genuinas e Impostoras.

En cada sistema biométrico hay un estricto compromiso entre la FMR y FNMR [Golfarelli et al., 1997]. De hecho, la FMR y la FNMR son funciones del valor umbral de decisión y deben ser referidos como FMR(t) y FNMR(t) respectivamente. Si t se reduce para hacer al sistema más tolerante con respecto a las variaciones de la entrada y el ruido, entonces la FMR(t) se incrementa y viceversa, si t se incrementa para hacer el sistema más seguro, entonces la FNMR(t) también se incrementa. Un diseñador del sistema puede no conocer de antemano la aplicación particular en que será usado (o un solo sistema puede diseñarse para una amplia variedad de aplicaciones). Entonces, es aconsejable reportar el desempeño del sistema en todos los puntos de operación, a través de la Curva Característica del Receptor (*Receiver Operating Characteristic* - ROC). Una curva ROC (ver figura 2.13) es una gráfica de la FMR versus (1-FNMR) para varios valores umbrales de decisión (a menudo la FNMR es reportada a lo largo del eje vertical en lugar de (1-FNMR)).

Además de las curvas y distribuciones anteriormente mencionadas, algunos indicadores "compactos" también son usados para resumir la precisión de un sistema de verificación (ver figura 2.12).

Tasa de Igual Error *(Equal Error Rate* - EER): Denota la tasa de error en el umbral $t = t_o$ para el cual la tasa de falsa coincidencia y la tasa de falsa no coincidencia son idénticas, es decir $FMR(t_0)=FNMR(t_0)$. En la práctica, puesto que las distribuciones de la puntuación de coincidencia no son curvas continuas (debido a que hay un número finito de pares comparados y a la cuantización de las puntuaciones de salida) no existe un punto medio EER exacto. En este caso, se debe reportar un intervalo en vez de un único valor [Maio et al., 2000]. Aunque la EER es un indicador importante, en la práctica un sistema biométrico basado en huellas dactilares raramente trabaja en el punto de operación correspondiente al EER, además se fija a menudo un valor umbral más severo para reducir la FMR a pesar del incremento de la FNMR.

ZeroFNMR: Es la FMR más baja en donde no ocurren falsas no coincidencias (ver figura 2.12).

ZeroFMR: Es la FNMR más baja en que no ocurren falsas coincidencias

(ver figura 2.12).

Tasa de Falla de Captura (*Failure to Capture* - FTC): Es el porcentaje de veces que el dispositivo biométrico falla al capturar automáticamente la biométrica cuando es presentada sobre la superficie del sensor. Una FTC alta hace que el dispositivo biométrico sea difícil de usar.

Tasa de Falla en la Inscripción (*Failure To Enroll* - FTE): Es el porcentaje de veces que los usuarios no pueden inscribirse en el sistema de reconocimiento. Hay un compromiso entre la tasa de FTE y la precisión (FMR y FNMR) de un sistema. Los errores de la FTE ocurren típicamente cuando el sistema de reconocimiento comprueba la calidad para asegurar que sólo las plantillas de buena calidad sean almacenadas en la base de datos. Como resultado la precisión del sistema (FMR y FNMR) mejora.

Tasa de Falla de Coincidencia (*Failure to Match* - FTM): Es el porcentaje de veces en que la entrada no puede ser procesada o coincidida con una plantilla válida, debido a la insuficiente calidad de la imagen de entrada. Esto es diferente al error por falsa no coincidencia, de hecho en un error de falla de coincidencia el sistema no puede tomar una decisión, mientras que en el error de falsa no coincidencia el sistema decide erróneamente que la entrada y la plantilla no son de la misma huella.

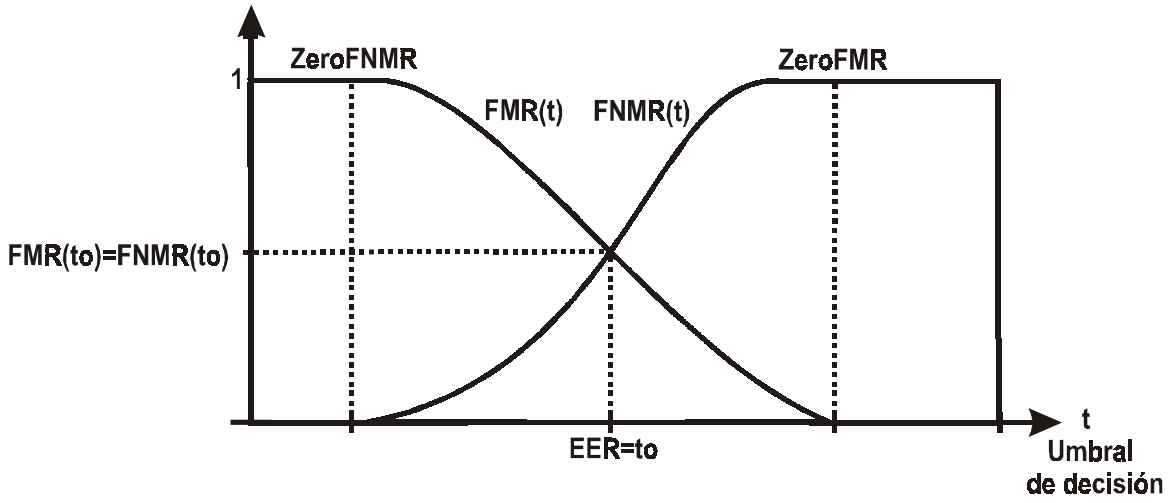


Figura 2.12 Gráfica de los indicadores FMR, FNMR, ZeroFMR, ZeroFNMR, EER respecto a los valores umbrales de decisión.

Los requerimientos prácticos del desempeño de un sistema biométrico están muy relacionados con la aplicación. Desde el punto de vista de la precisión del sistema, una FNMR extremadamente baja puede ser el objetivo inicial. Por ejemplo, en algunas aplicaciones forenses como en identificación criminal, la FNMR genera mayor preocupación que la FMR, es decir no queremos extraviar a un delincuente incluso con el riesgo de examinar manualmente un gran número de coincidencias potenciales identificadas por el sistema biométrico. En dichas aplicaciones, el experto humano es quien toma la última decisión. Al otro extremo, en una aplicación de control de acceso de alta seguridad en dónde el objetivo primario es evitar el ingreso de impostores, una muy baja FMR puede ser lo más idóneo, aunque estamos comprometidos con la posible molestia a los usuarios legítimos debido a la alta FNMR. Entre estos extremos hay varias aplicaciones civiles donde ambas tasas necesitan ser consideradas. Por ejemplo, para las tarjetas de verificación para el acceso a redes, una falsa coincidencia significa una pérdida de varios cientos de dólares, mientras que una alta FNMR puede irritar a los clientes. La figura 2.13 muestra gráficamente el compromiso preferido entre la FMR y FNMR para diferentes tipos de aplicaciones.

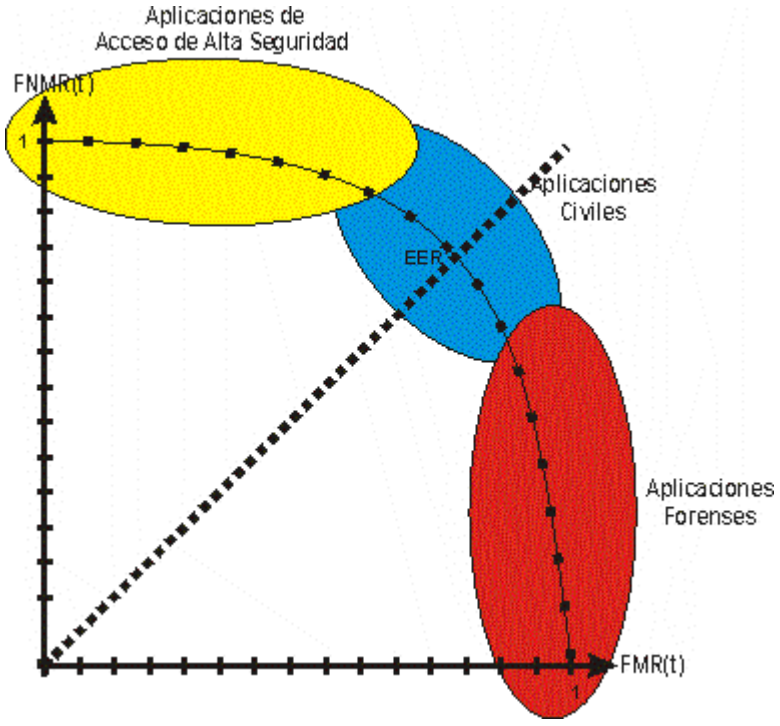


Figura 2.13 Puntos de operación típicos para diferentes aplicaciones mostradas en la curva ROC.

2.5.2. En sistemas de identificación

Tomando algunas consideraciones por simplificación, una estimación del desempeño en el modo de identificación puede ser inferida a través del error estimado en el modo de verificación. Asumiendo que no existe ningún mecanismo de indexado ni recuperación disponible (es decir, la búsqueda se realiza en toda la base de datos de N plantillas) y que cada usuario presenta una sola plantilla en la base de datos. Si denotamos a las FMR_N y $FNMR_N$ como la tasa de falsa coincidencia y la tasa de falsa no coincidencia respectivamente, entonces:

$FNMR_N$ = FNMR. De hecho, la probabilidad de falsa no coincidencia es igual que en el modo de verificación (excepto que esta expresión no toma en cuenta la probabilidad de que una falsa coincidencia pueda ocurrir antes de que la plantilla correcta sea visitada [Cappelli et al., 2000]).

$FMR_N = 1-(1- FMR)^N$. De hecho, una falsa coincidencia ocurre cuando la entrada coincide falsamente con una o más plantillas en la base de datos. La FMR_N se calcula entonces como uno menos la probabilidad que ninguna falsa coincidencia sea hecha con cualquiera de las plantillas de la base de datos. En la expresión anterior (1-FMR) es la probabilidad que la entrada no coincida falsamente con una sola plantilla y $(1-FMR)^N$ es la probabilidad que no coincida falsamente cualquiera de las plantillas en la base de datos. Si la FMR es muy pequeña, entonces la expresión anterior puede ser aproximada por FMR_N=N.FMR y por consiguiente podemos afirmar que la probabilidad de falsa coincidencia aumenta linealmente con el tamaño de la base de datos.

Este resultado tiene serias implicaciones para el diseño de los sistemas de identificación a gran escala. Usualmente la velocidad de cálculo es percibida como el más grande problema al escalar una aplicación de identificación. Realmente la precisión al escalar empeora la velocidad; de hecho considere una aplicación de identificación con 10 000 usuarios. Ciertamente podemos encontrar una combinación de un algoritmo rápido más una arquitectura rápida (eventualmente aprovechándose del paralelismo) capaz de realizar una identificación en algunos segun-

dos. Por otro lado, suponga que para una FNMR aceptable, la FMR del algoritmo escogido es 10^{-5} (es decir, sólo habrá una falsa coincidencia por cada 100 000 coincidencias). Entonces la probabilidad de aceptar a un individuo falsamente durante la identificación genera aproximadamente un FNMR de 10% y cada individuo tiene una buena oportunidad de acceder al sistema intentando ingresar los diez dedos de sus manos. Los sistemas biométricos multimodales parecen ser la única solución obvia a la precisión en la escalabilidad para la identificación automática a gran escala.

Si en la base de datos las plantillas han sido clasificadas/indexadas, entonces durante la identificación se realiza la búsqueda sólo en una porción de ésta, produciendo a su vez una formulación diferente de las $FNMR_N$ y FMR_N, es decir:

$FNMR_N$ = RER + (1-RER).FNMR. Donde la Tasa de Error de Recuperación (*Retrieval Error Rate* - RER) es la probabilidad de que la plantilla de la base de datos que corresponde a la huella buscada sea descartada erróneamente por el mecanismo de recuperación. La expresión anterior es obtenida usando siguiente argumento: en caso de que la plantilla no se recupere correctamente (esto sucede con probabilidad RER), el sistema siempre genera una falsa no coincidencia, considerando el caso en donde la recuperación retorna la plantilla correcta (esto sucede con probabilidad 1-RER), la tasa de falsas no coincidencias del sistema es FNMR. También, esta expresión es sólo una aproximación y no considera la probabilidad de coincidir falsamente una plantilla incorrecta antes que la correcta sea recuperada [Cappelli et al, 2000].

$FMR_N = 1- (1-FMR)^{N.PR}$. Donde la Tasa de Penetración (*Penetration Rate* - PR) es el porcentaje promedio de la base de datos buscada durante la identificación de una huella de entrada.

2.6. Aplicaciones

Pueden ser divididas principalmente en los siguientes grupos (ver figura 2.14):

Comerciales: Tales como el acceso a redes de computadoras, seguridad de

datos electrónicos, comercio electrónico, acceso a Internet, redes ATM, tarjetas de crédito, control de acceso físico, teléfonos celulares, PDA, mantenimiento de registros médicos y aprendizaje a distancia.

Gubernamentales: Tales como documentos de identidad nacional, licencias de conducir, seguridad social, control en fronteras y del pasaporte.

Forenses: Identificación de cadáveres, investigaciones criminales, identificación de terroristas, de niños y ancianos extraviados, entre otras.

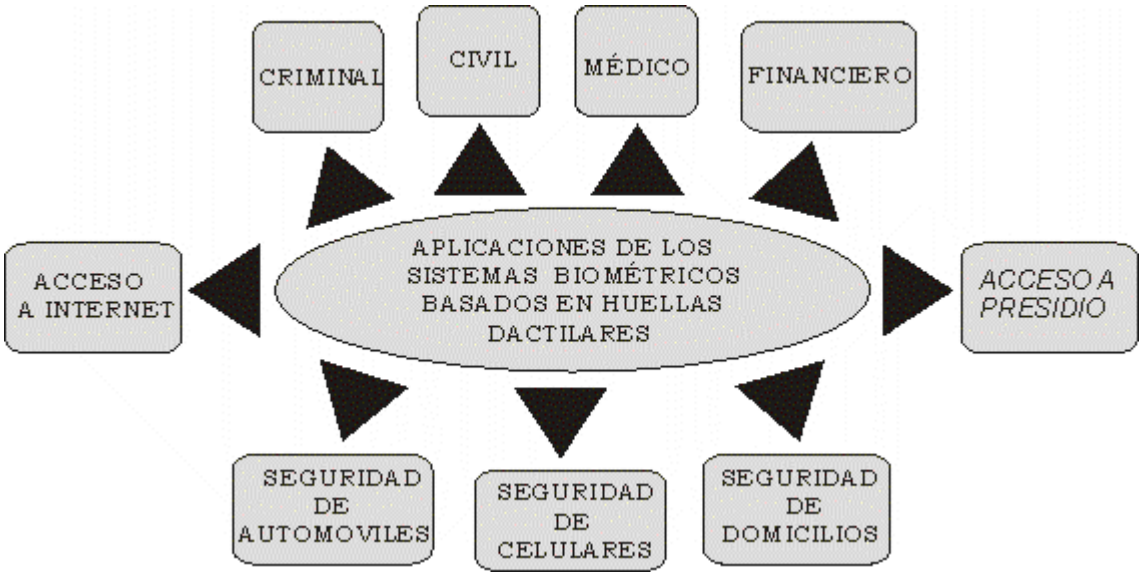


Figura 2.14 Aplicaciones de los sistemas biométricos basados en huellas dactilares.

2.7. Resumen

En este capítulo se ha presentado una descripción general de los sistemas biométricos especificando la arquitectura que presentan, sus modos de operación, sus medidas estándar de rendimiento y las tecnologías biométricas en las que se sustentan; en especial se detallan las características de los sistemas basados en huellas dactilares, exponiendo la problemática y las principales consideraciones de diseño. Los conceptos definidos en este capítulo son de suma importancia pues permiten la comprensión de los capítulos posteriores.

Capítulo III
Análisis de la Impresión Dactilar

3.1. Introducción

Este capítulo se inicia describiendo el cálculo de la Imagen de Orientación, un paso de suma importancia para el reconocimiento por huellas dactilares porque resume la información de la orientación de las crestas.

Por otro lado se describe el cálculo de la Imagen de Frecuencias, el cual es esencial para el mejoramiento, la extracción de características y la coincidencia de impresiones dactilares, pues resume la información acerca del período de las crestas.

Finalmente, se describe el cálculo de la Imagen de Regiones, que permite separar el área donde están presentes las crestas/valles del área corrompida que no provee información útil para el reconocimiento, evitando así la extracción de falsas características en áreas ruidosas de la impresión dactilar.

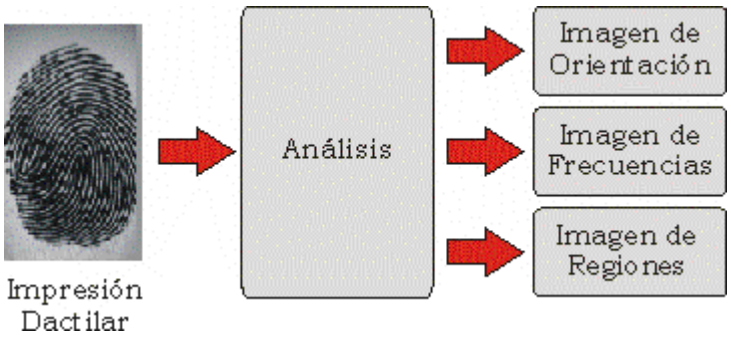


Figura 3.1 Etapa de análisis de la impresión dactilar.

Antes de realizar una explicación detallada, es necesario establecer ciertas pautas. A lo largo de este trabajo una imagen digital será representada matemáticamente por una función bidimensional discreta o matriz de puntos **I** de tamaño

$N^{I}_{Fil} x N^{I}_{Col}$, cuyo valor en el punto (u, v) denominado píxel (*picture element*) representa la iluminación o intensidad de luz del píxel de la u-ésima fila y la v-ésima columna (ver figura 3.2), éste es uno de los posibles valores cuantizados a ocho bits, que en escala de grises van desde 0 (negro) a 255 (blanco).

Los componentes del Vector Gradiente sobre cada punto (u, v) de la imagen $\mathbf{I}$ están representados por las matrices $\mathbf{G}_x$ y $\mathbf{G}_y$, ambas del mismo tamaño que $\mathbf{I}$.

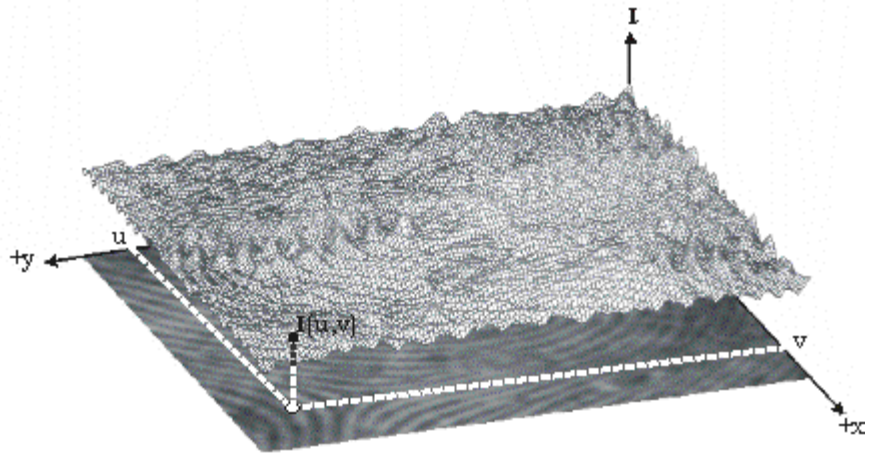


Figura 3.2 Representación de la imagen como una superficie tridimensional o como una matriz de puntos (píxeles) que varían en el rango de 0 a 255 (escala de grises).

3.2. Imagen de Orientación

La orientación local de una cresta en el píxel (u, v) de la imagen $\mathbf{I}$, es el ángulo que se forma entre la cresta perteneciente a una pequeña vecindad centrada en (u, v) y el eje horizontal de la imagen (ver figura 3.3). Las orientaciones locales son en principio perpendiculares a los Vectores Gradiente, por lo que podrán ser derivadas a partir de éstos. La Imagen de Orientación Θ también conocida como Imagen o Campo Direccional, es definida como una matriz de tamaño $N_{Fil} x N_{Col}$, donde $\Theta(i, j)$ representa la orientación promedio normal a las crestas del bloque (i, j) de $\mathbf{I}$, tal que $N_{Fil} \leq N^{I}_{Fil}$ y $N_{Col} \leq N^{I}_{Col}$.

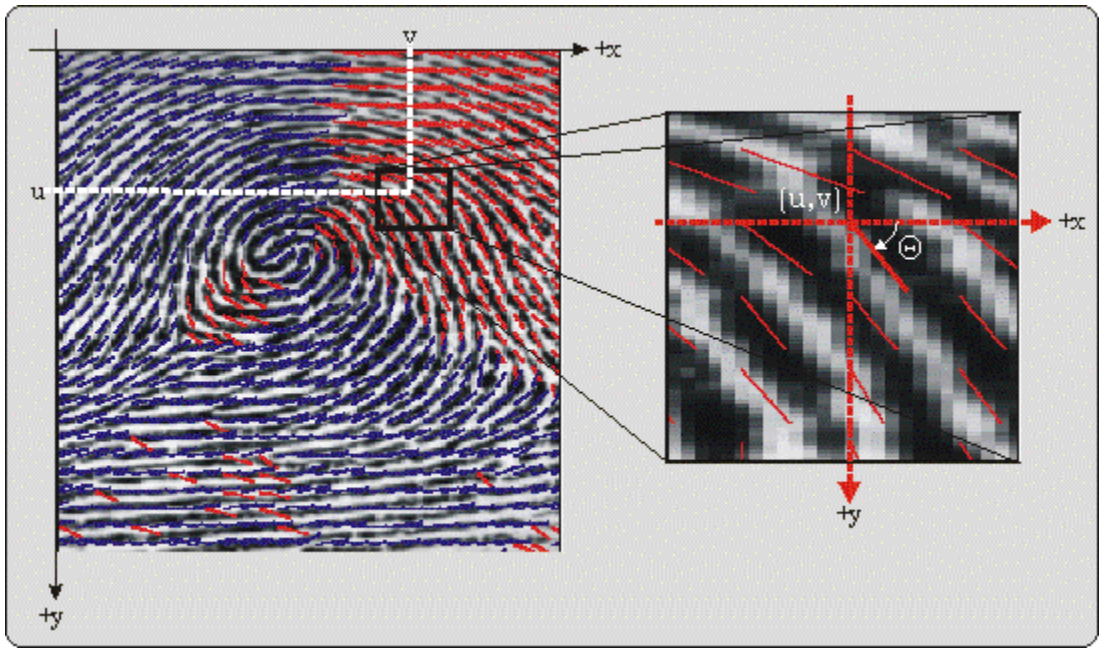


Figura 3.3 Imagen de Orientación de una impresión dactilar, cada línea representa la orientación promedio de las crestas en una vecindad de tamaño definido. Imagen 23_1.tif de la base de datos 2 del FVC2002.

3.2.1. Obtención de la Imagen de Orientación

La manera más simple y natural para determinar la Imagen de Orientación está basada en el uso del Vector Gradiente, para ello el algoritmo requiere realizar los siguientes pasos:

1. **Cálculo del Vector Gradiente**: Determinar los componentes del Vector Gradiente $\mathbf{G}_x$ y $\mathbf{G}_y$.
2. **Sectorización rectangular de la imagen**: Especificar el número de bloques y el tamaño de cada bloque en los que se va a dividir la imagen.
3. **Estimación de la orientación local**: Calcular la orientación local promedio referida a cada bloque (i, j).
4. **Suavizamiento**: Atenuar las perturbaciones presentes en la Imagen de Orientación estimada.

A continuación veremos en detalle cada uno de estos pasos, mostrando sólo el cálculo del Vector Gradiente en el anexo C.

3.2.1.1. Sectorización rectangular de la imagen

La división por bloques de una imagen y la obtención de un valor representativo en cada bloque, implica una disminución dimensional de los datos. Si se tiene una imagen de tamaño NxN y se divide en bloques de tamaño $N_k x N_k$ (en [Lobato et al, 2000] y [Lobato y Seara, 2000] se recomienda $N_k = 2^k$), se generan $(N/N_k)^2$ valores representativos, es decir se obtiene una reducción de N_k^2 datos acelerando así todo procesamiento posterior. Para el caso de $\mathbf{I}$, la división en bloques de tamaño $N_k x N_k$ producirá una Imagen de Orientación Θ de tamaño $N_{Fil} x N_{Col}$, donde $N_{Fil} = \lfloor N^I_{Fil}/N_k \rfloor$ y $N_{Col} = \lfloor N^I_{Col}/N_k \rfloor$ (ver figura 3.4). Si denotamos por $\mathbf{S}^k_{ij}$, al (i, j) -ésimo bloque vinculado a $\mathbf{I}$ de tamaño $N_k x N_k$, las posiciones en la imagen $\mathbf{I}$ pertenecientes a dicho bloque serán:

$$\mathbf{S}^k_{ij} = \{(u,v)/(i-1).N_k \le u \le i.N_k - 1,\ (j-1).N_k \le v \le j.N_k - 1\} \qquad (3.1)$$

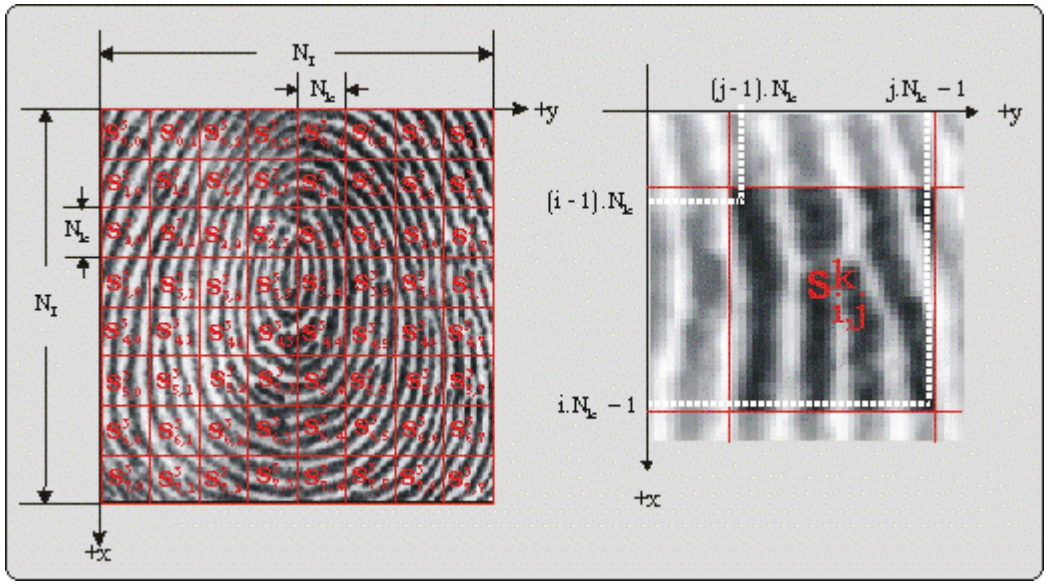


Figura 3.4 Sectorización de la imagen en bloques rectangulares.

Como veremos en capítulos posteriores, no únicamente es posible utilizar una sectorización rectangular, sino también es viable otro tipo de grillas, como las circulares especializadas para la tarea de reconocimiento.

3.2.1.2. Estimación local de la orientación

En esta sección calculamos la orientación dominante de cada bloque (i, j) utilizando el promediado del Vector Gradiente al cuadrado propuesto por [Bazen y Gerez, 2000] y [Lobato et al., 2000], así como también a través de la descomposición en valores singulares (*Singular Value Decomposition* - SVD) propuesto por [Guang y Milanfar, 2002]. En ambos algoritmos presentamos una mejora incluyendo una etapa de filtrado para la supresión del ruido.

3.2.1.2.1. Basada en el promediado del Vector Gradiente al cuadrado

Los Vectores Gradiente no pueden ser simplemente promediados en alguna vecindad local, puesto que los vectores opuestos se cancelarían aunque tengan la misma orientación, debido a que las estructuras de crestas y valles locales no cambian cuando las rotamos en 180°. Una solución a este problema es duplicar los ángulos de los Vectores Gradiente antes de la promediación, de tal forma que los vectores opuestos apuntarán hacia la misma dirección y por consiguiente se reforzarán, mientras que los vectores perpendiculares se cancelarán.

Después de la promediación, los Vectores Gradiente tienen que ser reconvertidos a una representación de ángulo único. La orientación dominante de las crestas y valles es entonces perpendicular a la dirección del promedio del Vector Gradiente. Esta técnica fue propuesta por [Kass y Witkin, 1987] y luego adaptada para la estimación de la Imagen de Orientación por [Ratha et al., 1995] y [Jain et al., 1997a]. El algoritmo propuesto por [Bazen y Gerez, 2000] no sólo duplica el ángulo sino también la magnitud del Vector Gradiente es elevada al cuadrado, considerándolo como un número complejo, esto produce que las orientaciones con una gran magnitud posean una alta importancia en la orientación promedio.

El primer paso a realizar es determinar el Vector Gradiente al cuadrado para cada píxel (u, v) a través de su representación compleja, es decir:

$$\begin{aligned}\mathbf{G}_{s,x}(u,v) + j.\mathbf{G}_{s,y}(u,v) &= \left(\mathbf{G}_x(u,v) + j.\mathbf{G}_y(u,v)\right)^2 \\ &= \left(\mathbf{G}_x^2(u,v) - \mathbf{G}_y^2(u,v)\right) + j\left(2.\mathbf{G}_x(u,v).\mathbf{G}_y(u,v)\right)\end{aligned} \tag{3.2}$$

Además se incluye una etapa de filtrado sobre los Vectores Gradiente al cuadrado para suprimir el ruido presente, utilizando un filtro Wienner basa bajos bidimensional [Lim, 1990, pp. 536 - 540] $\mathbf{W}_{WN}$ de tamaño $N_{W_{WN}} x N_{W_{WN}}$, obteniéndose lo siguiente:

$$\mathbf{G}'_{s,x}(u,v) = \mathbf{G}_{s,x}(u,v) * \mathbf{W}_{WN}(u,v) \tag{3.3}$$

$$\mathbf{G}'_{s,y}(u,v) = \mathbf{G}_{s,y}(u,v) * \mathbf{W}_{WN}(u,v) \tag{3.4}$$

Luego, se calcula el promedio del Vector Gradiente al cuadrado vinculado a cada bloque $\mathbf{S}_{ij}^k$, siendo:

$$\overline{\mathbf{G}}_{s,x}(i,j) = \sum_{u=(i-1).N_k}^{i.N_k-1} \sum_{v=(j-1).N_k}^{j.N_k-1} \mathbf{G}'_{s,y}(u,v) \tag{3.5}$$

$$\overline{\mathbf{G}}_{s,y}(i,j) = \sum_{u=(i-1).N_k}^{i.N_k} \sum_{v=(j-1).N_k}^{j.N_k} \mathbf{G}'_{s,y}(u,v) \tag{3.6}$$

Finalmente, calculamos la orientación promedio del Vector Gradiente al cuadrado $\mathbf{\Phi}(i,j)$ a través de la relación:

$$\mathbf{\Phi}(i,j) = \frac{1}{2}\operatorname{arctg2}(\overline{\mathbf{G}}_{s,x}(i,j), \overline{\mathbf{G}}_{s,y}(i,j)) \quad \text{tal que,} \quad -\frac{\pi}{2} \le \mathbf{\Phi}(i,j) \le \frac{\pi}{2} \tag{3.7}$$

Donde $\operatorname{arctg2}(x,y)$ es el arco tangente de y/x teniendo en cuenta el signo de x e y, cuya definición es:

$$\operatorname{arctg2}(x,y) = \begin{cases} \tan^{-1}(y/x) & x \ge 0 \\ \tan^{-1}(y/x) + \pi & x < 0 \wedge y \ge 0 \\ \tan^{-1}(y/x) - \pi & x < 0 \wedge y < 0 \end{cases} \tag{3.8}$$

3.2.1.2.2. Basada en la Descomposición en Valores Singulares

La técnica presentada está basada en el uso de la Descomposición en Valores Singulares (*Singular Value Decomposition* - SVD) sobre el conjunto de Vectores Gradiente pertenecientes a cada vecindad local. La SVD es una potente herramienta matemática que presenta una variedad de aplicaciones [Leach, 2000]. Esta técnica fue propuesta por [Guang y Milanfar, 2002] que incluye además una etapa de reducción del ruido a través de un proceso multiescala.

La determinación de la orientación dominante en una vecindad local puede ser reformulada como la tarea de hallar un vector unitario $\mathbf{a}_\phi = [\cos(\phi), \mathrm{sen}(\phi)]^T$ que minimice la función de costo:

$$J(\phi) = \sum_{u=(i-1).N_k}^{i.N_k-1} \sum_{v=(j-1).N_k}^{j.N_k-1} (\mathbf{a}_\phi^{\mathbf{T}}.\nabla\mathbf{I}(u,v))^2 \text{ tal que: } \nabla\mathbf{I}(u,v)=[\mathbf{G}_x(u,v), \mathbf{G}_y(u,v)]^T \qquad (3.9)$$

Ésta representa la energía asociada al promedio de la suma de los cosenos del ángulo formado entre $\mathbf{a}_\phi$ y cada $\nabla\mathbf{I}(u,v)$ perteneciente al bloque $\mathbf{S}_{ij}^k$. Si expresamos la función de costo en forma matricial, obtendremos:

$$J(\phi) = (\mathbf{G}_{ij}^{xy}.\mathbf{a}_\phi)^T.(\mathbf{G}_{ij}^{xy}.\mathbf{a}_\phi) = \mathbf{a}_\phi^{\mathbf{T}}.(\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}).\mathbf{a}_\phi \qquad (3.10)$$

Tal que:

$$\mathbf{G}_{ij}^{xy} = \begin{bmatrix} \nabla\mathbf{I}((i-1).N_k, (j-1).N_k) \\ \nabla\mathbf{I}((i-1).N_k+1, (j-1).N_k+1) \\ \vdots \\ \nabla\mathbf{I}(i.N_k-1, j.N_k-1) \end{bmatrix}_{N_k^2 x 2} \qquad (3.11)$$

La matriz $\mathbf{G}_{ij}^{xy}$ es la agrupación de los Vectores Gradiente pertenecientes al bloque $\mathbf{S}_{ij}^{k}$, dentro de una matriz $N_k^2 x2$. Es demostrable que $J(\phi)$ asume su mínimo valor cuando $\mathbf{a}_\phi$ es el eigenvector de la matriz $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$ correspondiente al más pequeño eingenvalor. Una de las técnicas más idóneas por ser numéricamente estable y eficiente para la determinación de dicho eigenvector es a través del cálculo de la SVD de $\mathbf{G}_{ij}^{xy}$, éste consiste en representar dicha matriz como el producto de tres matrices, es decir:

$$\mathbf{G}_{ij}^{xy} = \mathbf{U}_{ij}.\mathbf{D}_{ij}.\mathbf{V}_{ij}^{T} \qquad (3.12)$$

Donde:

$\mathbf{U}_{ij}$: Es una matriz ortogonal $N_k^2 x N_k^2$, tal que sus columnas contienen los eigenvectores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$.

$\mathbf{V}_{ij}$: Es una matriz ortogonal 2x2, tal que sus columnas contienen los eigenvectores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$.

$\mathbf{D}_{ij}$: Es una matriz diagonal $N_k^2 x 2$, cuyos elementos de la diagonal son las raíces cuadradas de los eingenvectores de $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$ o $\mathbf{G}_{ij}^{xy}.\mathbf{G}_{ij}^{xy^T}$ y se organizan en orden descendente.

La primera columna de $\mathbf{V}_{ij}$ es el menor eigenvector de la matriz $\mathbf{G}_{ij}^{xy^T}.\mathbf{G}_{ij}^{xy}$, por lo que la orientación dominante $\Phi(i, j)$ se obtiene a través de la relación:

$$\Phi(i, j) = \text{arctg2}(\mathbf{V}_{ij}(1,1), \mathbf{V}_{ij}(2,1)) \quad \text{tal que,} \quad -\frac{\pi}{2} \le \Phi(i, j) \le \frac{\pi}{2} \qquad (3.13)$$

La figura 3.5 muestra el promedio estimado del Vector Gradiente, apreciándose que las orientaciones dominantes estimadas para cada una de las vecindades (rectángulos verdes) son normales a la orientación que presenta toda estructura de crestas y valles.

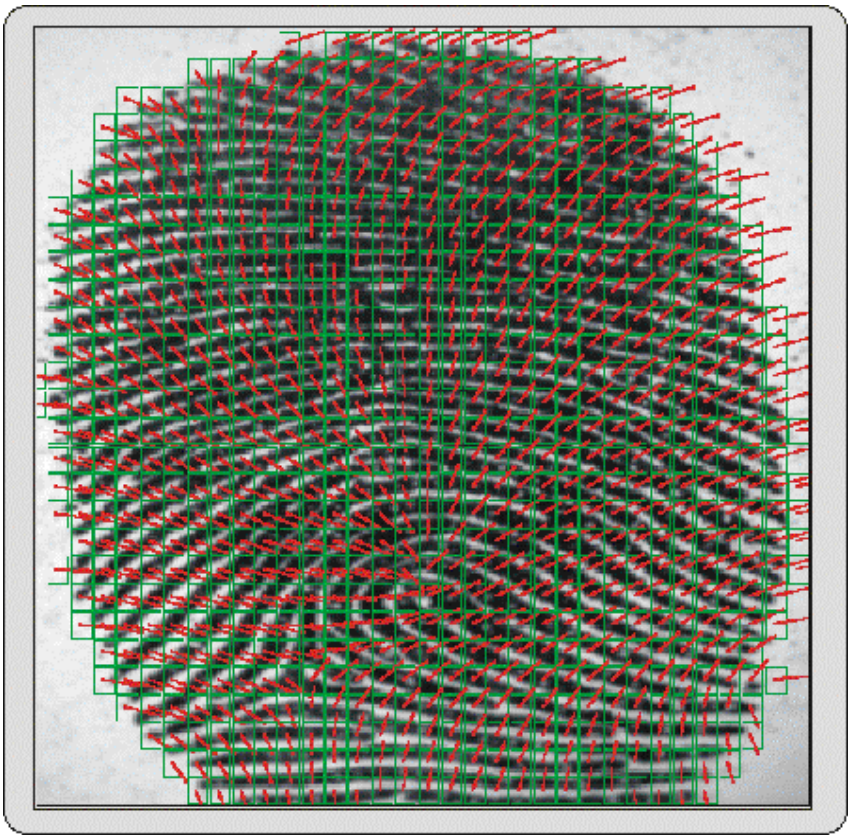


Figura 3.5 Promedio del Vector Gradiente Φ para vecindades específicas. Imagen 40_6.tif de la base de datos 2 del FVC2002.

3.2.1.3. Suavizamiento

Es un paso necesario para proveer una mayor robustez en la estimación de la Imagen de Orientación, por lo que se han propuesto diferentes técnicas para la realización de esta operación, dos de las cuales serán presentadas aquí. La primera técnica propuesta por [Karu y Jain, 1996] filtra los componentes sinusoidales del Vector Gradiente promediado y reestima las orientaciones a través del índice de consistencia. La segunda técnica propuesta por [Perona, 1999] utiliza el concepto de difusión para realizar la operación de filtrado. A continuación veremos en detalle cada una de ellas y las mejoras que hemos propuesto.

3.2.1.3.1. Basado en el filtrado de los componentes seno-coseno

Puesto que $\Phi(i, j)$ varía lentamente en una vecindad local en donde no aparezcan puntos singulares, un filtro pasa bajo puede usarse para corregir $\Phi(i, j)$, lo cual se requiere convertir a $\Phi(i, j)$ en un campo vectorial continuo, veamos:

$$\gamma_x(i, j) = \cos(2.\Phi(i, j)) \tag{3.14}$$

$$\gamma_y(i, j) = \text{s en}(2.\Phi(i, j)) \tag{3.15}$$

Donde $\gamma_x(i, j)$ y $\gamma_y(i, j)$ son los componentes x e y del campo vectorial. Con el campo vectorial resultante el filtrado pasa bajo puede ser realizado como sigue:

$$\gamma'_x(i, j) = \gamma_x(i, j) * \mathbf{W}_{SH}(i, j) \tag{3.16}$$

$$\gamma'_y(i, j) = \mathbf{W}_{SH}(i, j) * \gamma_y(i, j) \tag{3.17}$$

Donde $\mathbf{W}_{SH}$ es la máscara de un filtro pasa bajo bidimensional de tamaño $N_{W_{SH}} x N_{W_{SH}}$. Esta operación es realizada a nivel de bloques por lo que no representa un elevado costo computacional. Para el cálculo de la orientación dominante a partir de los componentes x e y del campo vectorial, se usa la siguiente expresión:

$$\Phi'(i, j) = \frac{1}{2}\text{arctg2}(\gamma'_x(i, j), \gamma'_y(i, j)) \tag{3.18}$$

Se vuelve a estimar el valor de $\Phi'(i, j)$ si algún parámetro de medida indica que las orientaciones en la vecindad del bloque $\mathbf{S}_{ij}^{k}$ están muy dispersas respecto a $\Phi'(i, j)$. Dicho parámetro de medida viene dado por el nivel de consistencia. Esta etapa fue planteada por [Jain et al., 1997a], sin embargo hemos variado la definición del nivel de consistencia expuesta en su trabajo de manera que este parámetro quede normalizado en el rango de 0 a 1.

3.2.1.3.1.1. Nivel de consistencia

El nivel de consistencia en la vecindad local al bloque $\mathbf{S}_{ij}^{k}$ se calcula a través de las siguientes relaciones:

$$\alpha(i_v, j_v, i, j) = (\boldsymbol{\Phi}'(i_v, j_v) - \boldsymbol{\Phi}'(i, j) + 2\pi) \bmod (2\pi);\ \forall \boldsymbol{\Phi}'(i_v, j_v) \neq 0;\ 0 \leq \alpha \leq 2\pi \quad (3.19)$$

$$\beta(i_v, j_v, i, j) = \begin{cases} \alpha(i_v, j_v, i, j) & 0 < \alpha(i_v, j_v, i, j) \leq \pi/2 \\ \left\| \alpha(i_v, j_v, i, j) - \pi \right\| & \pi/2 < \alpha(i_v, j_v, i, j) \leq 3\pi/2 \\ 2\pi - \alpha(i_v, j_v, i, j) & 3\pi/2 < \alpha(i_v, j_v, i, j) \leq 2\pi \end{cases} ;\ 0 \leq \beta \leq \pi/2 \quad (3.20)$$

$$\mathbf{C}_o(i, j) = \frac{1}{\pi N_c} \sqrt{\sum_{i_v = i - \bar{N}_c}^{i + \bar{N}_c} \sum_{j_v = j - \bar{N}_c}^{j + \bar{N}_c} \beta^2(i_v, j_v, i, j)} \ ;\ \bar{N}_c = \left\lceil \frac{N_c}{2} \right\rceil ;\ 0 \leq \mathbf{C}_o \leq 1 \quad (3.21)$$

Donde N_c es el tamaño de la vecindad local alrededor del bloque $\mathbf{S}_{ij}^{k}$. $\boldsymbol{\Phi}(i, j)$ y $\boldsymbol{\Phi}(i_v, j_v)$ son las orientaciones dominantes en los bloques $\mathbf{S}_{ij}^{k}$ y $\mathbf{S}_{i_v j_v}^{k}$ respectivamente. La relación (3.19) asegura que $\alpha(i_v, j_v, i, j)$ se encuentre entre $\leq 0, 2\pi \geq$, tal como muestra figura 3.6.

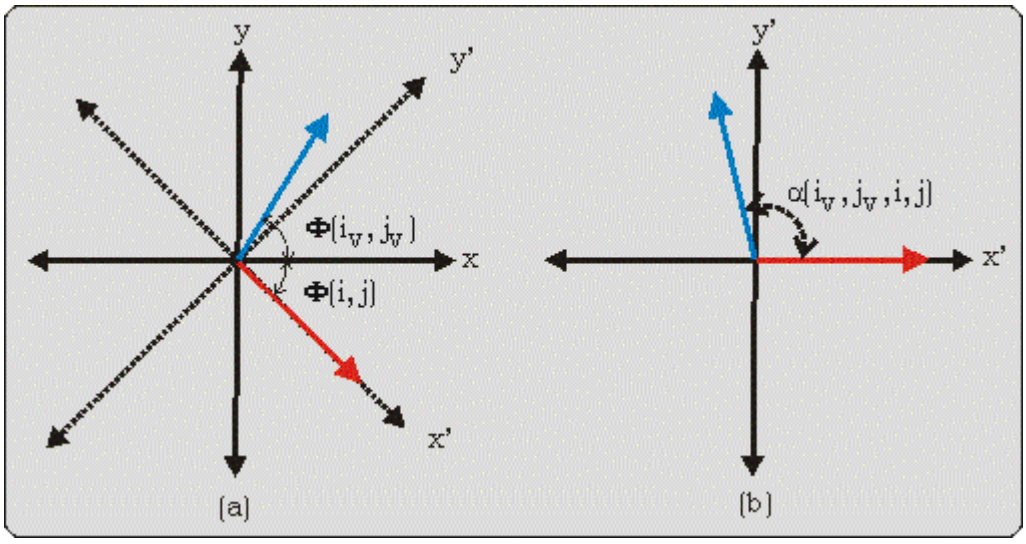


Figura 3.6 (a) Orientaciones dominantes en los bloques $\mathbf{S}_{ij}^{k}$ y $\mathbf{S}_{i_v j_v}^{k}$; (b) diferencia entre las orientaciones dominantes $\alpha(i_v, j_v, i, j)$.

La relación (3.20) traslada al primer cuadrante a todos los $\alpha(i_v, j_v, i, j)$ que se encuentren entre $\leq \pi/2, 2\pi \geq$ siendo esta operación necesaria ya que las contribuciones realizadas por los $\alpha(i, j, u, v)$ que se encuentren entre $\leq \pi/2, 2\pi \geq$ son equivalentes a las contribuciones realizadas por los $\alpha(i_v, j_v, i, j)$ ubicados entre $\leq 0, \pi/2 \geq$, tal como se muestra en la figura 3.7.

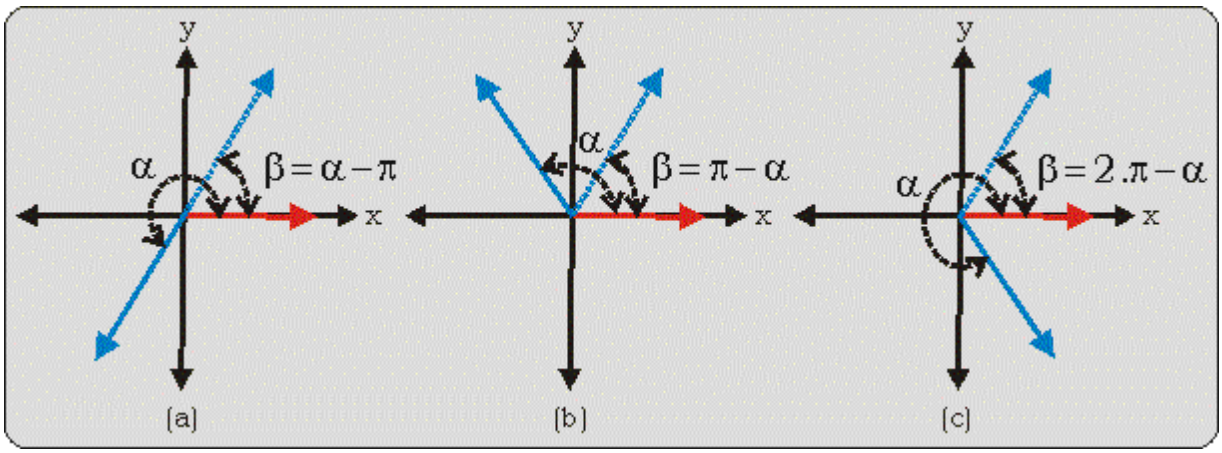


Figura 3.7 Reducción de α al primer cuadrante; (a) del III al I cuadrante; (b) del II al I cuadrante; (c) del IV al I cuadrante.

La relación (3.21) define el elemento (i, j) de la Matriz de Consistencia $\mathbf{C}_o$ como el error cuadrático promedio de las diferencias angulares respecto al elemento central $\mathbf{\Phi}'(i, j)$ en la vecindad de tamaño $N_c x N_c$. Bajo esta definición $\mathbf{C}_o(i, j)$ presentará dos comportamientos extremos:

1. El primero se presenta cuando todas las orientaciones son iguales a la orientación que presenta el elemento central $\mathbf{\Phi}'(i, j)$, por lo que $\beta(i, j, u, v) = 0;\ \forall\ (u, v) \in N_c x N_c$, implica que: $\mathbf{C}_o(i, j) = 0$.
2. El segundo caso se presenta cuando todas las orientaciones son contrarias a la orientación que presenta el elemento central $\mathbf{\Phi}'(i, j)$, por lo que $\beta(i, j, u, v) = \pi;\ \forall\ (u, v) \in N_c x N_c$ implica que: $\mathbf{C}_o(i, j) = 1$.

Entre estas dos situaciones extremas, $\mathbf{C}_o(i, j)$ variará en el rango de 0 y 1 (ver figura 3.9), por lo tanto provee los requerimientos de medida. En la figura 3.8 se muestran todos los pasos requeridos para el cálculo del nivel consistencia.

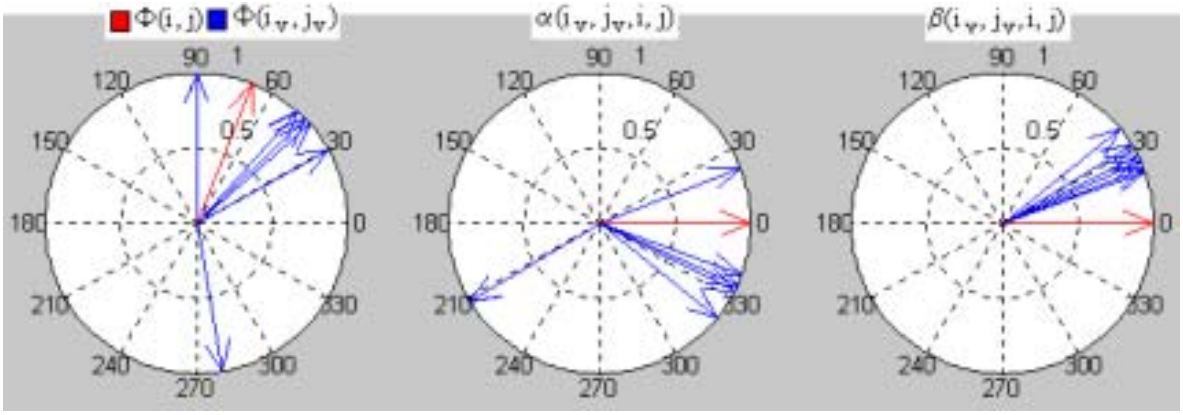


Figura 3.8 Pasos para el cálculo del nivel de consistencia en una vecindad.

Si el nivel de consistencia es menor que un valor umbral, entonces $\Phi'(i, j)$ es reestimada para un nivel más bajo de resolución hasta que el nivel de consistencia esté sobre el valor del umbral.

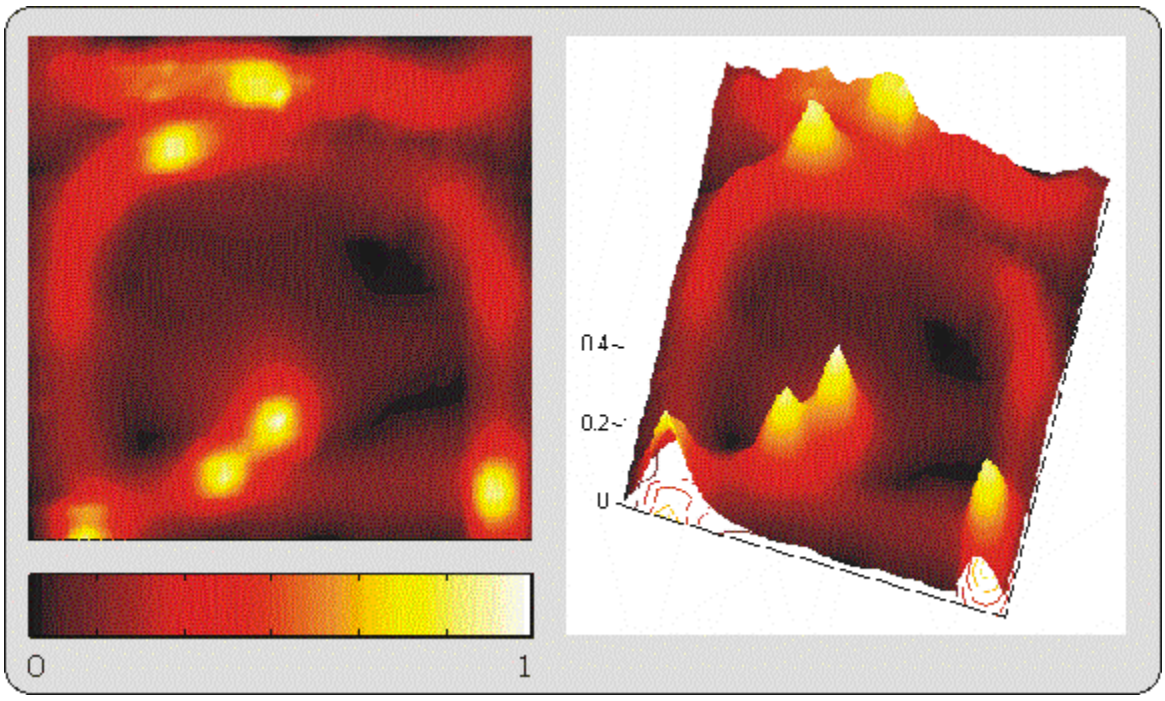


Figura 3.9 Gráfica de la Matriz de Consistencia $\mathbf{C}_o$ de la imagen 40_6.tif de la base de datos 2 del FVC2002.

Luego se procede a determinar la Imagen de Orientación $\Theta(i, j)$, tomando en cuenta que es perpendicular a $\Phi'(i, j)$.

$$\Theta(i,j) = \begin{cases} \Phi'(i,j) + \dfrac{\pi}{2} & \Phi'(i,j) \le 0 \\ \Phi'(i,j) - \dfrac{\pi}{2} & \Phi'(i,j) > 0 \end{cases} \tag{3.22}$$

Tal que, $-\frac{\pi}{2} \le \Theta(i,j) \le \frac{\pi}{2}$. La figura 3.10 muestra la Imagen de Orientación antes y después del filtrado seno - coseno y la reestimación a través del nivel de consistencia.

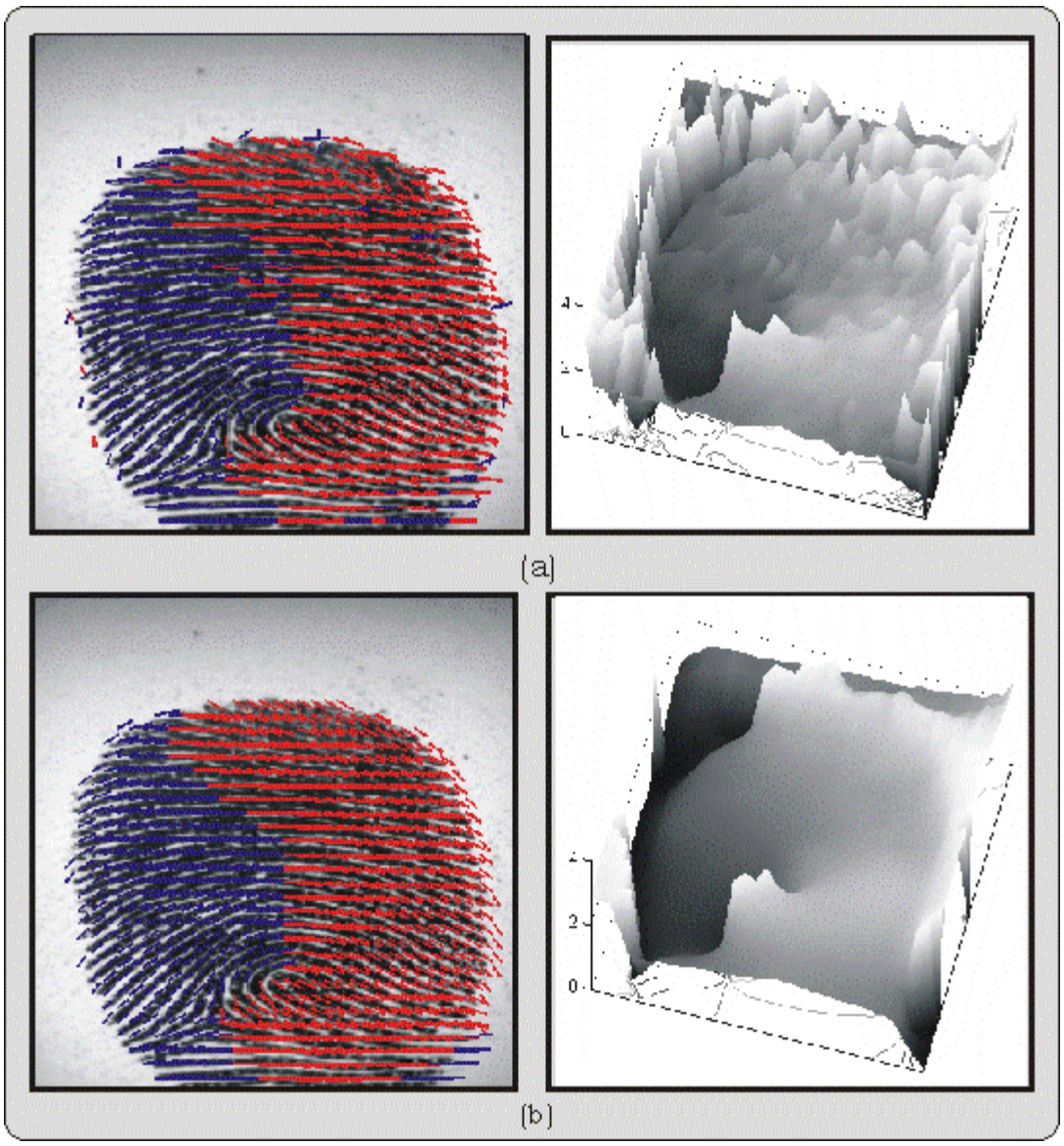


Figura 3.10 Suavizamiento de la Imagen de Orientación a través del filtrado de los componentes seno-coseno (a) antes de filtrar; (b) después de filtrar. Imagen 40_6.tif de la base de datos 2 del FVC2002.

3.2.1.3.2. Basado en difusión

La difusión es una importante técnica para el procesamiento de imágenes y visión computacional debido a que provee una conveniente manera de suavizar el ruido presente en la imagen. Este modelo se explica a partir de la descripción del transporte de la materia en los procesos de difusión isotrópico, descritos por:

$$u(k,0) = u_0(k) \tag{3.23}$$

$$\frac{\partial u(k,t)}{\partial t} = \lambda \frac{\partial^2 u(k,t)}{\partial k^2} \tag{3.24}$$

Donde:

u: Es la concentración de la solución o el medio que varía únicamente en una dirección específica.

k: Es la dirección única a lo largo de la cual varía el medio.

λ: Es la velocidad de difusión.

Debido a que se desea suavizar las orientaciones promedio de las crestas y valles para cada uno de los bloques la ecuación de difusión será:

$$\theta_p(k,0) = \Theta(i,j) \quad ; p = i + (j-1).N_k \tag{3.25}$$

$$\frac{\partial \theta_p(k,t)}{\partial t} = \lambda \frac{\partial^2 \theta_p(k,t)}{\partial k^2} \tag{3.26}$$

Donde $\Theta(i,j)$ es la orientación promedio de las crestas y valles en el bloque (i,j), calculado a partir de la ecuación (3.22) sin haber realizado ninguna operación de suavizado sobre $\Phi(i,j)$. La energía vinculada a cada observación $\theta_p(k,t)$, viene dada por la función de Lagrange:

$$L(t,k,\theta_p,\theta_p^{(1)}) = T(\theta_p^{(1)}(k,t)) - U(\theta_p(t,k)) \tag{3.27}$$

Donde:

$\theta_p^{(1)}(k,t) = \dfrac{\partial \theta_p(k,t)}{\partial k}$: Es la derivada parcial de la observación $\theta_p(k,t)$ respecto a la dirección de desplazamiento k.

T: Es la energía cinética debido a la variación de la observación $\theta_p(k,t)$ a medida que se desplaza en la dirección de desplazamiento k.

U: Es la energía potencial debida a la presencia de las observaciones vecinas a $\theta_p(k,t)$.

L: Es la función de Lagrange, que representa la energía total que posee la observación $\theta_p(k,t)$ a medida que se desplaza en la dirección de desplazamiento k bajo la influencia de las observaciones vecinas.

El proceso de suavizado de la orientación consiste en la minimización de la energía promedio vinculada a la observación $\theta_p(k,t)$ a medida que ésta se desplaza en la dirección de desplazamiento k. Esto es:

$$\varepsilon(t) = \int_{k_1}^{k_2} L(t,k,\theta_p,\theta_p^{(1)})dk \qquad (3.28)$$

Del cálculo de variaciones [O'Neil, 1994, pp 569-586] sabemos que $\varepsilon(t)$ tomará un comportamiento extremal (será mínimo) cuando la función de Lagrange cumpla con la ecuación de Euler, es decir:

$$\frac{\partial L}{\partial \theta_p} - \frac{\partial}{\partial k}\left(\frac{\partial L}{\partial \theta_p^{(1)}}\right) = 0 \qquad (3.29)$$

implicará:

$$\frac{dU}{d\theta_p} + \frac{\partial^2 \theta_p}{\partial k^2} = 0 \qquad (3.30)$$

tal que:

$$T = \frac{1}{2}\left(\frac{\partial \theta_p}{\partial k}\right)^2 \qquad (3.31)$$

De las ecuaciones (3.26) y (3.30) se obtiene:

$$\frac{\partial \theta_p(k,t)}{\partial t} = -\lambda \frac{dU(\theta_p(k,t))}{d\theta_p} \tag{3.32}$$

La representación discreta de la ecuación (3.32) es:

$$\theta_p(k+1,t) = \theta_p(k,t) - \lambda \frac{dU(\theta_p(k,t))}{d\theta_p} \tag{3.33}$$

La ecuación (3.32) nos permite caracterizar el proceso de difusión a través de la energía potencial producida por la interacción de la observación en cuestión con sus vecinos. Ahora cabe señalar que la función U representará con eficacia las interacciones con las observaciones vecinas, de modo que al transcurrir el tiempo, los valores de orientación queden uniformizados. Es decir, si la observación θ_p está muy dispersa respecto a sus vecinos, U deberá tomar un valor alto, lo que originará un incremento positivo en la observación θ_p estimada; si θ_p está muy cercana a sus vecinos, U deberá tomar un valor bajo, lo que originará un incremento negativo en θ_p. Para ambos casos la energía U deberá depender de alguna métrica que permita cuantificar la distancia de θ_p frente a sus vecinos, podría suponerse que la métrica adecuada sería la diferencia aritmética de dos ángulos cualesquiera, pero tal definición presenta ciertas ambigüedades debido al período 2π que presentan los ángulos. Por ejemplo, si tenemos dos ángulos que toman los valores $\theta_A = 0$ y $\theta_B = \pi/2$, la distancia entre éstos tendrá dos valores $\pi/2$ y $-3\pi/2$ (ver figura 3.11.a). Una métrica que supera este inconveniente es la distancia Euclidiana entre los vectores unitarios relacionados a cada una de las orientaciones (ver figura 3.11.b). Para la utilización de esta métrica se requiere representar cada observación θ_p sobre un círculo unidad, es decir vincular a cada θ_p con un vector unitario que tenga su misma orientación, tal como se indica en [Perona, 1999].

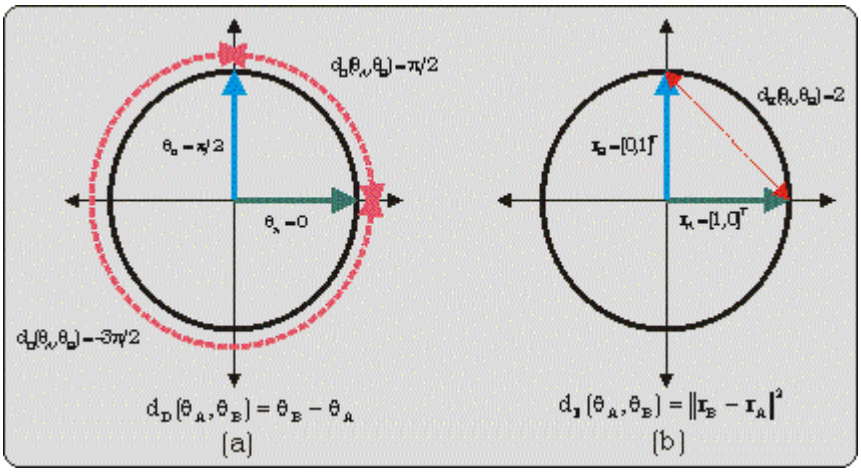


Figura 3.11 Representaciones de la distancia entre dos ángulos (a) métrica definida como la diferencia entre dos ángulos; (b) métrica definida como la distancia Euclidiana entre dos vectores unitarios, vinculado cada vector a una respectiva orientación.

Bajo esta perspectiva, una función de energía recomendable y planteada por [Perona, 1999] es la suma de las distancias Euclidianas normalizadas al cuadrado de cada uno de los vectores vinculados con las observaciones vecinas respecto al vector vinculado a θ_p, dicha métrica es denotada por $d_E(\theta_\upsilon, \theta_p)$. Permitiendo definir la función de energía como:

$$U_E(\theta_p(t,k)) = \sum_{\upsilon \in V(p)} d_E(\theta_\upsilon, \theta_p) = \sum_{\upsilon \in V(p)} \frac{1}{2} \left\| \mathbf{r}_\upsilon - \mathbf{r}_p \right\|^2 = 2. \sum_{\upsilon \in V(p)} \text{sen}^2\left(\frac{\theta_\upsilon - \theta_p}{2}\right) \tag{3.34}$$

Donde: $\mathbf{r}_\upsilon = [\cos(\theta_\upsilon), \text{sen}(\theta_\upsilon)]^T$ y $\mathbf{r}_p = [\cos(\theta_p), \text{sen}(\theta_p)]^T$ y $V(p)$ es el conjunto de orientaciones vecinas a θ_p.

Utilizando esta métrica es obvio que a medida que θ_υ esté más alejado de θ_p, $d_E(\theta_\upsilon, \theta_p)$ será mayor. Sería adecuado variar la métrica de manera tal que se dé mayor peso a las observaciones más alejadas. Para tal propósito planteamos el uso de la distancia de Mahalanobis entre la observación θ_p con cada uno de sus vecinos, denotada por $d_M(\theta_\upsilon, \theta_p)$ y definida a través de la relación:

$$d_M(\theta_\upsilon, \theta_p) = \frac{1}{2}(\mathbf{r}_\upsilon - \mathbf{r}_p)^T.\mathbf{C}.(\mathbf{r}_\upsilon - \mathbf{r}_p) \tag{3.35}$$

donde, $\mathbf{C} = \mathbf{\Phi}^T.\mathbf{\Lambda}.\mathbf{\Phi}$, siendo:

$$\mathbf{\Phi} = \begin{bmatrix} \cos(\theta_p) & -\operatorname{sen}(\theta_p) \\ \operatorname{sen}(\theta_p) & \cos(\theta_p) \end{bmatrix} \tag{3.36}$$

$$\mathbf{\Lambda} = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \tag{3.37}$$

Reemplazando (3.36) y (3.37) en (3.31) obtenemos:

$$d_M(\theta_\upsilon, \theta_p) = \left((\lambda_1 + \lambda_2) + (\lambda_2 - \lambda_1).\cos(\theta_\upsilon + 3\theta_p)\right).\operatorname{sen}^2\left(\frac{\theta_\upsilon - \theta_p}{2}\right) \tag{3.38}$$

Tomando en todo momento $\lambda_1 = 1$, λ_2 será el parámetro que nos permita darle un mayor peso a las orientaciones más distantes respecto a θ_p. En la figura 3.12, podemos ver las gráficas de $d_E(\theta_\upsilon, \theta_p)$ y $d_M(\theta_\upsilon, \theta_p)$ para varios valores de λ_2. Tomando $\lambda_2 = 1$, $\theta_p = 0$ y $\theta_\upsilon \in \leq -\pi/2, \pi/2 \geq$. Se puede apreciar que para valores de θ_υ en el rango de $\leq -\pi/9, \pi/9 \geq$ los valores de $d_E(\theta_\upsilon, \theta_p)$ y $d_M(\theta_\upsilon, \theta_p)$ son cercanos y para los valores de θ_υ fuera de este rango el incremento es muy pronunciado, tal como se esperaba.

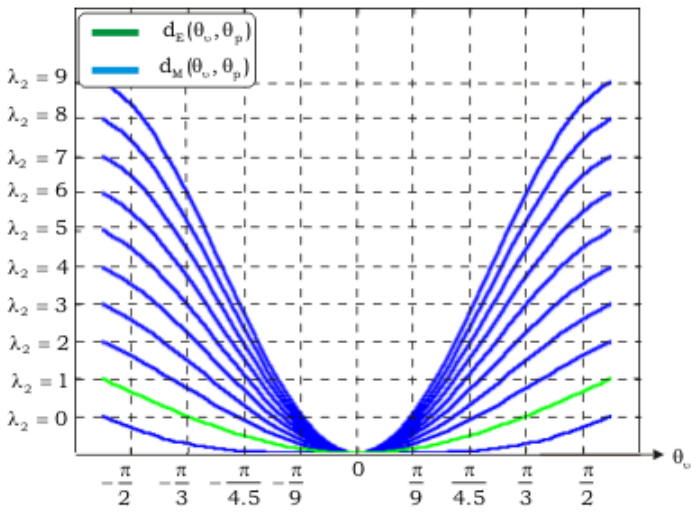


Figura 3.12 Gráficas de $d_E(\theta_\upsilon, \theta_p)$ y $d_M(\theta_\upsilon, \theta_p)$ para varios valores de λ_2.

Luego de haber definido la función de energía que caracteriza la interacción de la observación con sus vecinos, es necesario utilizar la representación discreta de la ecuación de difusión pues ésta es la que realiza el proceso de filtrado por difusión isotrópica, cada observación estimada es una función exclusiva de la observación anterior y donde el único vínculo existente con las observaciones originales es la condición inicial. Cabe destacar que cada vez que se incrementa el valor de λ se prolonga el tiempo de difusión, obteniéndose un mayor suavizado de las observaciones originales. La tabla 3.1 muestra las ecuaciones de difusión discretizadas para cada una de las métricas planteadas y la figura 3.13 muestra la difusión de la Imagen de Orientación de una impresión dactilar para cada métrica, considerando una vecindad de tamaño 5x5 píxeles, velocidad de difusión $\lambda = 0.01$, los parámetros de la distancia de Mahalanobis son: $\lambda_1 = 1.3$, $\lambda_2 = 1.9$ y cinco épocas de difusión. Note que θ_p es la Imagen de Orientación suavizada para cada epoca de difusión.

MÉTRICAS	ECUACIONES DE DIFUSIÓN DISCRETAS
EUCLIDIANA	$\theta_p(k+1,t) = \theta_p(k,t) - \lambda \sum_{\upsilon \in V(p)} \left\{ \frac{\partial d_E(\theta_\upsilon, \theta_p)}{\partial \theta_p} \right\}$
	$\frac{\partial d_E(\theta_\upsilon, \theta_p)}{\partial \theta_p} = -\text{sen}(\theta_\upsilon - \theta_p)$
MAHALANOBIS	$\theta_p(k+1,t) = \theta_p(k,t) - \lambda \sum_{\upsilon \in V(p)} \left\{ \frac{\partial d_M(\theta_\upsilon, \theta_p)}{\partial \theta_p} \right\}$
	$\frac{\partial d_M(\theta_\upsilon, \theta_p)}{\partial \theta_p} = -\frac{1}{2}(\lambda_1 + \lambda_2).\text{sen}(\theta_\upsilon - \theta_p) + \frac{1}{2}(\lambda_1 - \lambda_2).(3.\text{sen}(3\theta_p + \theta_\upsilon) - \text{sen}(2\theta_p + 2\theta_\upsilon) - 2.\text{sen}(4\theta_p))$

Tabla 3.1 Ecuaciones de difusión para cada una de las métricas planteadas.

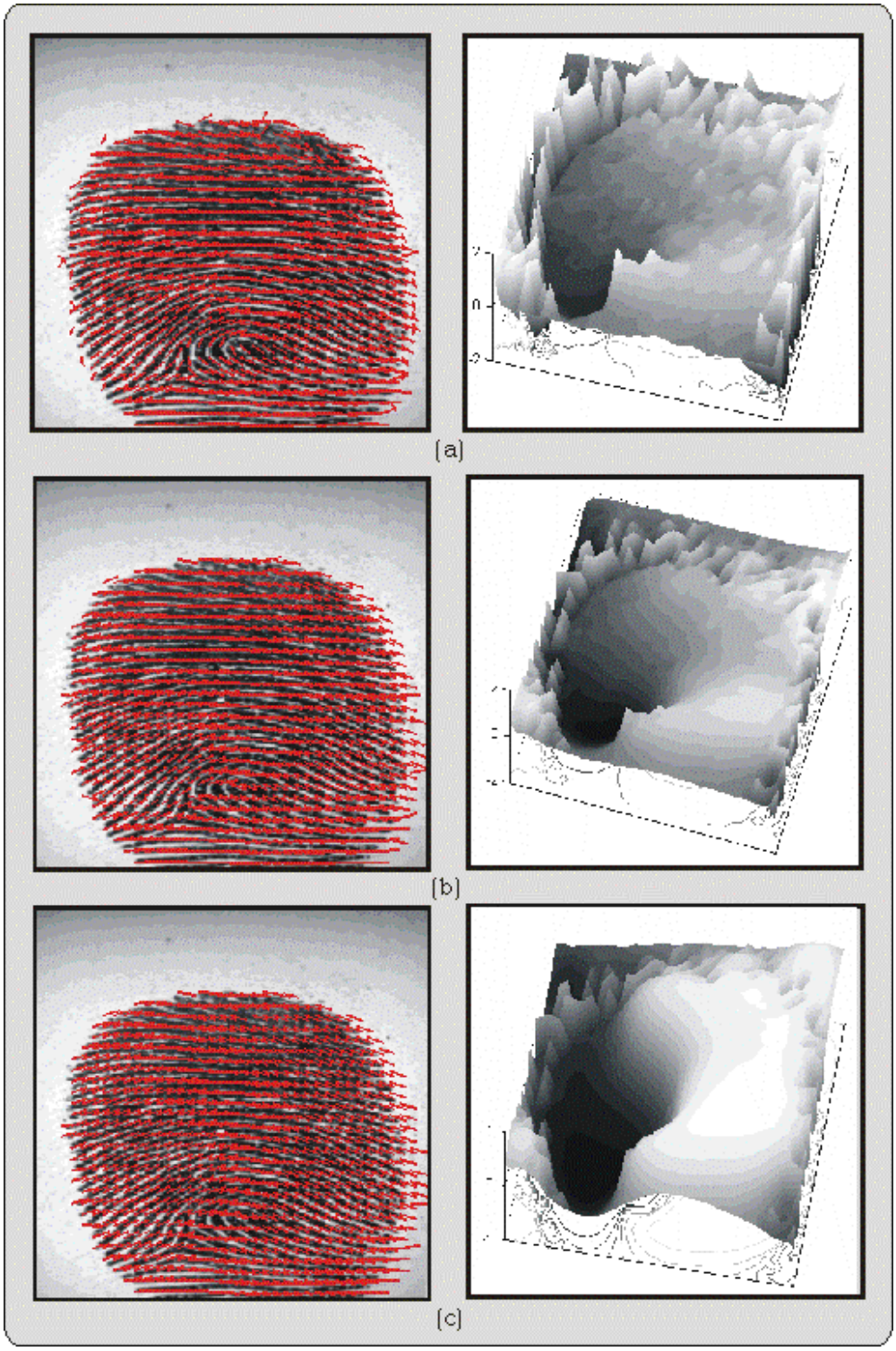


Figura 3.13 Suavizamiento de la Imagen de Orientación a través de difusión (a) antes de difuminar; (b) después de difuminar cinco veces consecutivas utilizando la distancia Euclidiana (c) después de difuminar cinco veces consecutivas utilizando la distancia de Mahalanobis. Imagen 40_6.tif de la base de datos 2 del FVC2002.

3.3. Imagen de Frecuencias

La frecuencia espacial local de las crestas o densidad en el punto (u, v) de la imagen $\mathbf{I}$ es la inversa del número de crestas por unidad de longitud a lo largo de un segmento centrado en (u, v) y ortogonal a la orientación de la cresta (ver figura 3.14). Esto es así, puesto que en una vecindad local (donde no estén presentes las minucias y los puntos singulares) los niveles de grises a lo largo de las crestas y valles pueden ser modelados como ondas sinusoidales de frecuencia espacial específica perpendiculares a la orientación de las crestas.

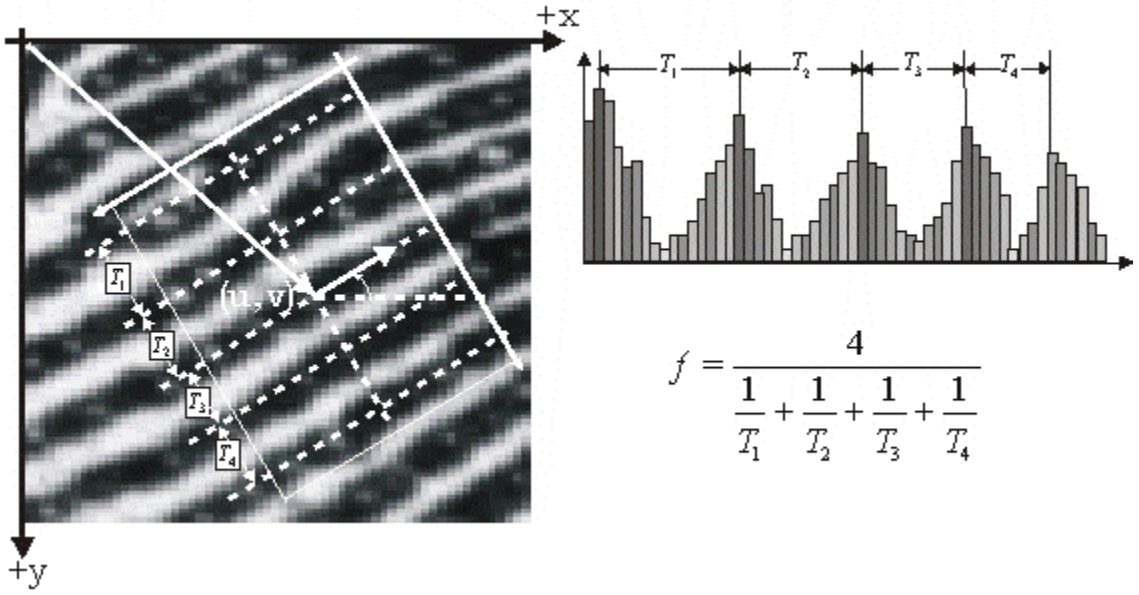


Figura 3.14 Definición de la frecuencia espacial en el píxel (u,v) a través de una vecindad local orientada según la dirección de las crestas.

De forma análoga a la Imagen de Orientación, la Imagen de Frecuencias se define como una matriz de tamaño $N_{Fil} x N_{Col}$, en donde $\Im(i,j)$ representa la frecuencia local de las crestas del bloque (i, j) de $\mathbf{I}$.

[Hong et al., 1998] estimaron la frecuencia espacial contando el número promedio de píxeles entre dos picos consecutivos de la signatura [Gonzales y Woods, 1992, pp.529-530] generada por la suma de los elementos de las columnas en una vecindad local orientada según la orientación de las crestas y valles. Es una técni-

ca simple y rápida, sin embargo es difícil detectar de forma confiable los picos consecutivos en aquellas vecindades ruidosas provenientes de la imagen.

[Maio y Maltoni, 1998] propusieron una técnica para estimar la Imagen de Frecuencias, modelando los patrones locales de las crestas como superficies sinusoidales y a través del teorema de la variación estimaron las frecuencias desconocidas. La variación de la función h en el intervalo $\leq x_1, x_2 \geq$ es la cantidad de cambio vertical respecto a h (ver figura 3.15):

$$\mathrm{Var}(h) = \int_{x_1}^{x_2} \left| \frac{\mathrm{d}h(x)}{\mathrm{d}x} \right| .\mathrm{d}x \qquad (3.39)$$

Si la función h es periódica o la amplitud sufre pequeños cambios en el intervalo $\leq x_1, x_2 \geq$, la variación puede ser expresada como una función de la amplitud promedio y la frecuencia promedio f, a través de la relación :

$$\mathrm{Var}(h) = (x_2 - x_1).2a_m.f \qquad (3.40)$$

Por consiguiente, la frecuencia no conocida puede ser estimada como:

$$f = \frac{\mathrm{Var}(h)}{(x_2 - x_1).2a_m} \qquad (3.41)$$

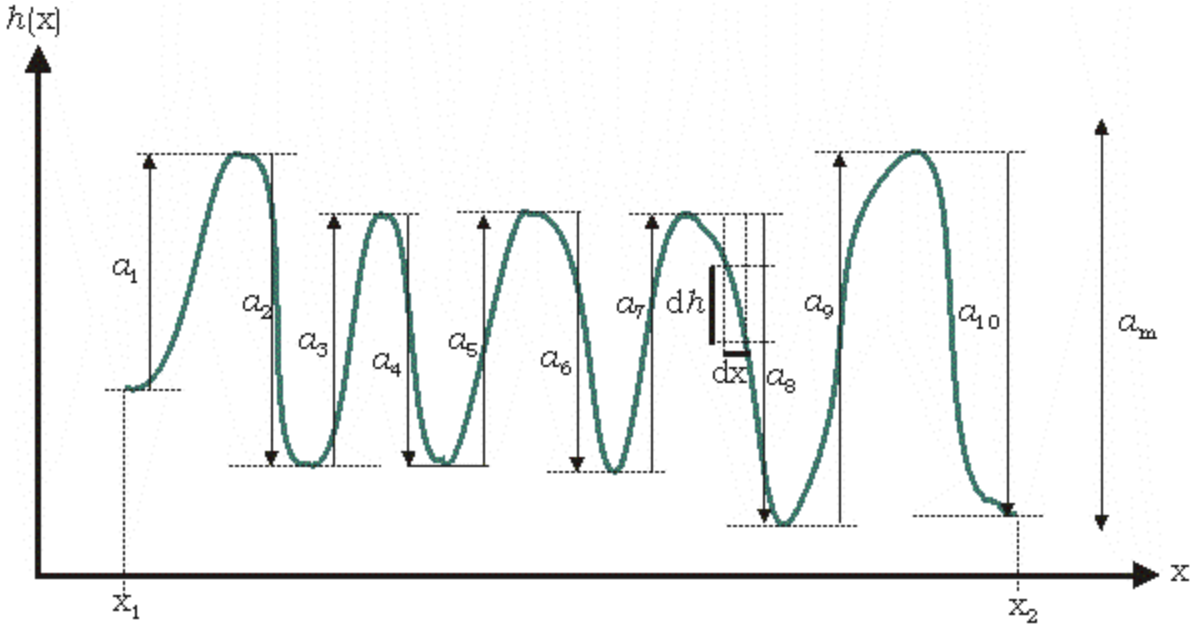


Figura 3.15 Variación en la amplitud de la función h en el intervalo $\leq x_1, x_2 \geq$.

3.3.1. Obtención de la Imagen de Frecuencias

El algoritmo para la obtención de la Imagen de Frecuencias presentado aquí, está basado en el trabajo de [Maio y Maltoni, 1998], cuya implementación a nivel discreto depende principalmente de las derivadas de primer y segundo orden (ver anexo C) de la imagen $\mathbf{I}$. Maio y Maltoni plantearon la siguiente relación basada en el teorema de variación para el cálculo de la frecuencia en el bloque (i, j).

$$\Im(i, j) = \frac{1}{2\pi} \cdot \sqrt{\frac{\Gamma_{xx}^2(i, j) + \Gamma_{yy}^2(i, j) + 2.\Gamma_{xy}(i, j)}{\Gamma_x^2(i, j) + \Gamma_y^2(i, j)}} \tag{3.42}$$

Donde:

$$\Gamma_{xx}(i, j) = \frac{\pi}{2}.\text{Mean}\left[\left|\mathbf{G}_{xx}(u_{cent}, v_{cent})\right|\right] \tag{3.43}$$

$$\Gamma_{yy}(i, j) = \frac{\pi}{2}.\text{Mean}\left[\left|\mathbf{G}_{yy}(u_{cent}, v_{cent})\right|\right] \tag{3.44}$$

$$\Gamma_{xy}(i, j) = \frac{\pi}{2}.\text{Mean}\left[\left|\mathbf{G}_{xy}(u_{cent}, v_{cent})\right|\right] \tag{3.45}$$

$$\Gamma_x(i,j) = \sqrt{2.\text{Mean}\left[\mathbf{G}_x^2(u_{cent}, v_{cent})\right]} \quad (3.46)$$

$$\Gamma_y(i,j) = \sqrt{2.\text{Mean}\left[\mathbf{G}_y^2(u_{cent}, v_{cent})\right]} \quad (3.47)$$

tal que: $u_{cent} = (i-1).N_k + \lfloor N_k/2 \rfloor$ y $v_{cent} = (j-1).N_k + + \lfloor N_k/2 \rfloor$

Donde:

$\mathbf{G}_{xx}$, $\mathbf{G}_{yy}$, $\mathbf{G}_{xy}$, $\mathbf{G}_x$ y $\mathbf{G}_y$: Son las derivadas de primer y segundo orden (Vector Gradiente y Matriz Hessiana).

Γ_{xx}, Γ_{yy}, Γ_{xy}, Γ_x y Γ_y: Son los valores ponderados de las respectivas derivadas calculados sobre una ventana centrada en el píxel (u_{cent}, v_{cent}) de tamaño $N_3 x N_3$ píxeles.

Las principales ventajas de esta técnica respecto a las anteriores son su robustez y sobretodo su bajo costo computacional, ya que se basa en el cálculo de un sólo operador y no en la heurística que pueda presentar la signatura de una vecindad orientada. En la figura 3.16 podemos ver la Imagen de Frecuencias obtenida con una ventana de tamaño 33x33 píxeles y aplicando un filtro de suavizamiento 5x5 píxeles.

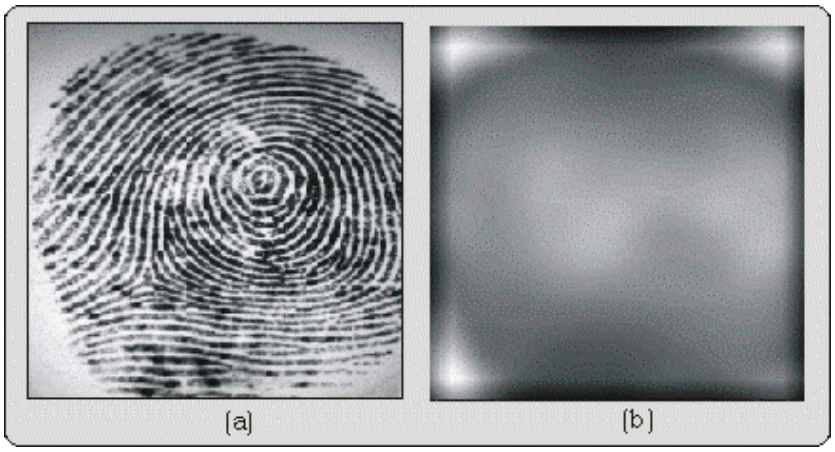


Figura 3.16. Estimación de la frecuencia espacial local de las crestas usando la técnica propuesta en [Maio y Maltoni, 1998] (a) impresión dactilar; (b) Imagen de Frecuencias. Imagen 18_1.tif de la base de datos 2 del FVC2002.

3.4. Imagen de Regiones

Generalmente una impresión dactilar está constituida por dos regiones de interés (ver figura 3.17). La región bien definida o primer plano y la región corrompida o segundo plano. En la primera región las estructuras de crestas y valles son claramente diferenciables y se forman del primer contacto de la yema de los dedos con la superficie del sensor, esta región es utilizada para el reconocimiento. En la segunda región las estructuras de crestas y valles no son claramente diferenciables debido a la presencia del ruido y las perturbaciones en la imagen tales como: bordes, manchas, arrugas, entre otras. Usualmente son formadas en áreas donde no hubo un contacto adecuado de la yema del dedo con la superficie del sensor, esta región es ignorada en el reconocimiento.



Figura 3.17 Regiones de una impresión dactilar. Imagen 99_1.tif de la base de datos 2 del FVC2002.

La tarea del algoritmo de segmentación es decidir que parte de la imagen corresponde al primer o segundo plano. Una correcta segmentación es especialmente importante para la confiable extracción de las características. La mayoría de algoritmos de extracción de características extraen una porción de falsas características cuando son aplicados al segundo plano. El principal objetivo del algoritmo de segmentación es desechar el segundo plano y por tanto reducir el número de falsas características, aumentando la robustez del sistema de reconocimiento.

Debido a que las huellas dactilares son patrones de crestas y valles, utilizar las técnicas clásicas de umbralización global o local [Gonzales y Woods, 1992, pp. 447-517] no permite aislar de forma efectiva el primer plano. En realidad lo que discrimina entre el primer y segundo plano no son las intensidades promedio de la imagen, sino la presencia de un patrón orientado en el primer plano y de un patrón isotrópico (es decir que no tiene una orientación dominante) en el segundo plano.

Varios acercamientos a la segmentación de impresiones dactilares son dados en la literatura: En [Mehtre et al ,1987] la impresión dactilar es sectorizada en bloques de 16x16 píxeles, calculándose para cada bloque el histograma de las orientaciones de las crestas. La presencia de un pico significativo en un histograma denota un patrón orientado (primer plano), un histograma aproximadamente plano representa un patrón uniforme (segundo plano). También [Mehtre y Chatterjee, 1989] propusieron una mejora sobre esta técnica. En [Ratha et al., 1995] la impresión dactilar es igualmente sectorizada a 16x16 píxeles por bloque, calculándose para cada bloque la varianza de los niveles de grises en la dirección normal a la orientación de la cresta, además derivaron de éste un índice de calidad. La idea principal es que en las regiones ruidosas no existe una dependencia direccional, pues el primer plano exhibe una alta varianza en una dirección normal a la orientación de las crestas y muy baja varianza en dirección de éstas. En [Maio y Maltoni, 1997] se sigue la misma tendencia sectorizando la impresión dactilar en bloques de 16x16 píxeles, determinándose el promedio de la magnitud del Vector Gradiente de cada bloque y puesto que una huella dactilar está compuesta por crestas y valles semi paralelas (orientadas), el Vector Gradiente tendrá una alta magnitud en el primer plano y un valor bajo en el segundo plano. En [Bazen y Gerez, 2000] se plantea una técnica de segmentación píxel por píxel usando la coherencia, cuando el nivel de coherencia para cada píxel es menor que un valor umbral se considera que pertenece al segundo plano, en caso contrario pertenecerá al primer plano, incluyéndose al final una etapa de post procesamiento morfológico usado para reducir las regiones mal clasificadas.

3.4.1. Obtención de la Imagen de Regiones

La Imagen de Regiones $\mathfrak{R}$, es definida como una imagen de tamaño $N^{I}_{Fil} x N^{I}_{Col}$, donde $\mathfrak{R}(u,v)$ indica una categoría relacionada con el píxel $\mathbf{I}(u,v)$. Un píxel (u,v) con valor 0 indica que el píxel $\mathbf{I}(u,v)$ correspondiente pertenece a una región no recuperable, en caso contrario (valor 1) $\mathbf{I}(u,v)$ pertenecerá a una región recuperable.

Como se comentó en la sección anterior, son dos las vertientes al momento de obtener las características para la segmentación de la imagen, la basada en bloques [Mehtre et al ,1987], [Mehtre y Chatterjee, 1989], [Ratha et al., 1995], [Maio y Maltoni, 1997] y la basada en píxeles [Bazen y Gerez, 2000]. A continuación se describen los algoritmos para ambas técnicas, incluyendo las mejoras que hemos realizado.

3.4.1.1. Basada en píxeles

La realización del algoritmo para la segmentación basada en píxeles presenta los siguientes pasos:

1. **Extracción de características**: Selección de las características en que se basará la segmentación, además de la obtención del conjunto de entrenamiento con la asistencia de un experto.
2. **Condensación del conjunto de entrenamiento**: Exclusión de las muestras de entrenamiento redundantes a través de un algoritmo de condensación de datos.
3. **Clasificación**: Definición del clasificador que catalogará a cada píxel como perteneciente al primer o segundo plano tomando como base el conjunto de entrenamiento condensado del paso anterior.
4. **Post procesamiento**: Reducción de las regiones mal clasificadas utilizando las herramientas de morfología matemática de imágenes binarias.

A continuación veremos en detalle cada uno de los pasos anteriormente indicados.

3.4.1.1.1. Extracción de características

El primer paso para el desarrollo del algoritmo de segmentación es la selección de las características o propiedades individuales de los píxeles que puedan contener información útil para la correcta discriminación entre ambas regiones. Tomando en cuenta los trabajos de [Bazen y Gerez, 2001] y [Bazen et al., 2001] seleccionamos tres características básicas: La coherencia, el valor promedio local y la varianza local. La coherencia y la varianza serán considerablemente altas en el primer plano, mientras que en el segundo plano el valor promedio es en general bajo. Para reducir los efectos del ruido de la imagen en cada píxel (u,v) las características son promediadas sobre una máscara Gaussiana $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$, siendo calculados el valor promedio local y la varianza local sobre el píxel (u,v) a través de las expresiones:

$$\text{Mean}[\mathbf{I}(u,v)] = \mathbf{I}(u,v) * \mathbf{W}_{GS}(u,v) \tag{3.48}$$

$$\text{Var}[\mathbf{I}(u,v)] = \text{Ener}[\mathbf{I}(u,v)] - \text{Mean}[\mathbf{I}(u,v)]^2 \tag{3.49}$$

Donde:

$$\text{Ener}[\mathbf{I}(u,v)] = \{\mathbf{I}(u,v)\}^2 * \mathbf{W}_{GS}(u,v) \tag{3.50}$$

El cálculo de la coherencia se realiza a través de la expresión (A.15). Estas tres características definen la observación asociada al píxel (u,v) de la imagen (ver figura 3.18), es decir:

$$\mathbf{x} = \left[\text{Mean}[\mathbf{I}(u,v)] \quad \text{Var}[\mathbf{I}(u,v)] \quad \text{Coh}[\mathbf{I}(u,v)]\right]^T ; \mathbf{x} \in \square^3 \tag{3.51}$$

Puesto que cada observación pertenecerá a una de las dos posibles clases (primer o segundo plano) denotadas por $\Omega = \{w_{-1}, w_1\}$, definimos un conjunto de aprendizaje o conjunto de entrenamiento $S_{\mathbf{x}}$.

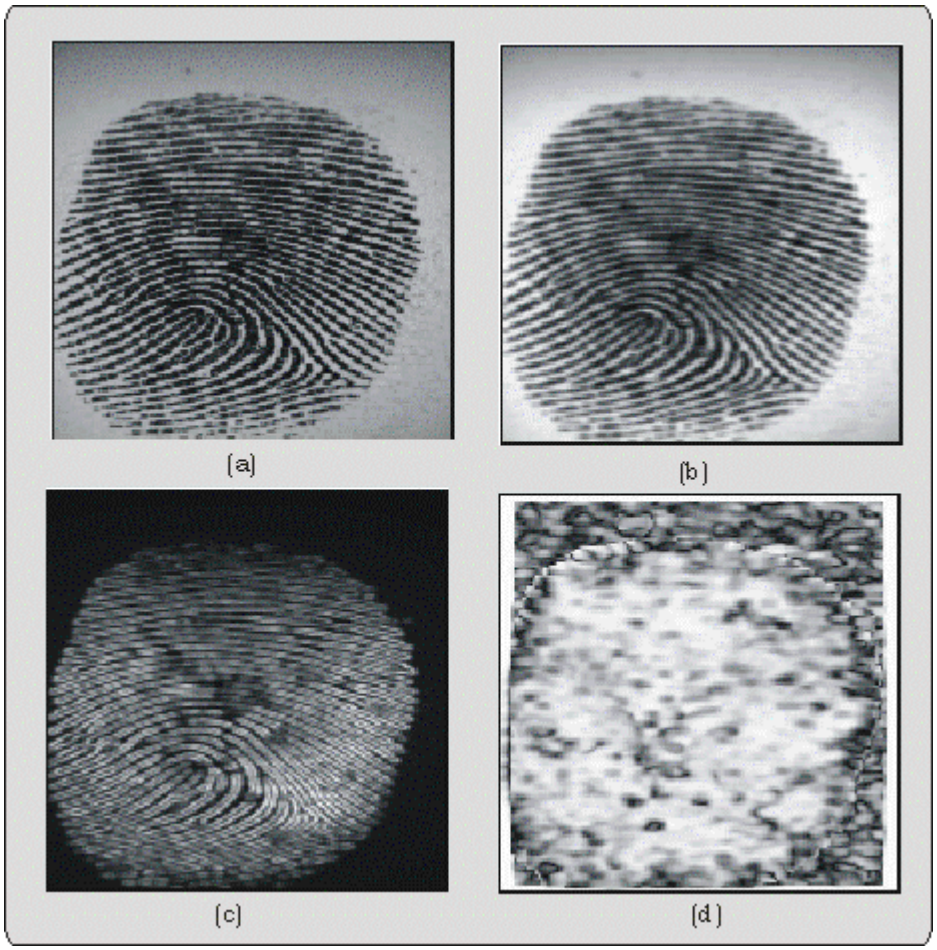


Figura 3.18 Características de los píxeles en una impresión dactilar (a) Impresión dactilar original; (b) valor promedio local; (c) desviación estándar local; (d) coherencia. Imagen 13_6.tif de la base de datos 2 del FVC2002.

Es evidente que el conjunto de entrenamiento $S_{\mathbf{x}}$ está compuesto por dos subconjuntos, en donde cada uno de ellos agrupa a todas las observaciones pertenecientes a una de las dos clases, es decir:

$$\begin{aligned} S_{\mathbf{x}} &= S_{\mathbf{x}_{-1}} \cup S_{\mathbf{x}_1} \\ &= \{(\mathbf{x}_j, c_j) \mid (\mathbf{x}_j \in \mathbf{X}_{-1} \vee \mathbf{x}_j \in \mathbf{X}_1) \wedge (c_j \in \Omega)\} \end{aligned} \tag{3.52}$$

Tal que: $j = 1, 2, .., n_{\mathbf{x}}$, siendo $n_{\mathbf{x}} = n_{\mathbf{x}_{-1}} + n_{\mathbf{x}_1}$, donde $n_{\mathbf{x}_{-1}}$ y $n_{\mathbf{x}_1}$ son el número de observaciones de $S_{\mathbf{x}_{-1}}$ y $S_{\mathbf{x}_1}$ respectivamente. Bajo esta perspectiva establecemos la siguiente notación:

$\mathbf{X}_k$: Denota todas las observaciones realizadas, vinculadas al conjunto de

entrenamiento $S_{\mathbf{x}_k}$. Es una matriz de tamaño $3xn_{\mathbf{x}_k}$, tal que $k = -1,1$.

$\mathbf{x}^i_{j:k}$: Denota el valor de la observación del conjunto de entrenamiento $S_{\mathbf{x}_k}$ para la característica i-ésima, tal que $i = 1,2,3$ y $j = 1,2,\cdots,n_{\mathbf{x}_k}$.

$\mathbf{x}_{j:k}$: Denota la j-ésima observación del conjunto de entrenamiento $S_{\mathbf{x}_k}$, es un vector columna 3x1 de la matriz $\mathbf{x}_k$.

$\mathbf{x}^i_k$: Denota los valores en las observaciones del conjunto de entrenamiento $S_{\mathbf{x}_k}$ para la i-ésima característica, es un vector fila $1xn_{\mathbf{x}_k}$ de la matriz $\mathbf{x}_k$.

Hasta este momento la determinación de las observaciones $\mathbf{x}_{j:k}$ está definida, pero no se ha especificado como establecer la clase a la que pertenece (w_{-1} ó w_1). Este es en si un proceso manual y realizado por un experto, puesto que utiliza su experiencia y conocimiento para realizar la segmentación. Lamentablemente no siempre el juicio humano es preciso, lo cual nos llevó a establecer el siguiente procedimiento para determinar una mejor segmentación manual:

1. Seleccionar un subconjunto de imágenes en la base de datos que servirán para generar cada conjunto de entrenamiento.
2. A través de un proceso manual (utilizando para ello cualquier programa editor de imágenes, en nuestro caso Corel 11) se obtiene para una imagen seleccionada, una imagen binaria vinculada, denominada imagen de regiones reconocibles (ver figura 3.19), que especifica el primer plano (blanco) y el segundo plano (negro).
3. Determinar las observaciones para cada píxel de la imagen seleccionada y generar los conjuntos de entrenamiento $S_{\mathbf{x}_{-1}}$ y $S_{\mathbf{x}_1}$.
4. Calcular las matrices $\mathbf{x}_{-1}$ y $\mathbf{x}_1$ vinculadas a cada una de las clases y determinar el histograma para cada $\mathbf{x}^i_k$; "observar" el grado de traslape que presentan los histogramas vinculados a los pares de características $\mathbf{x}^i_{-1}$ y $\mathbf{x}^i_1$ (ver figura 3.20).
5. Si el traslape es muy marcado entonces volvemos al paso 2, si el problema persiste la imagen es desechada y se analiza la siguiente imagen seleccionada, en caso contrario las muestras de entrenamiento obtenidas de la imagen son aceptadas. El proceso continúa hasta que todas imágenes seleccionadas son analizadas.

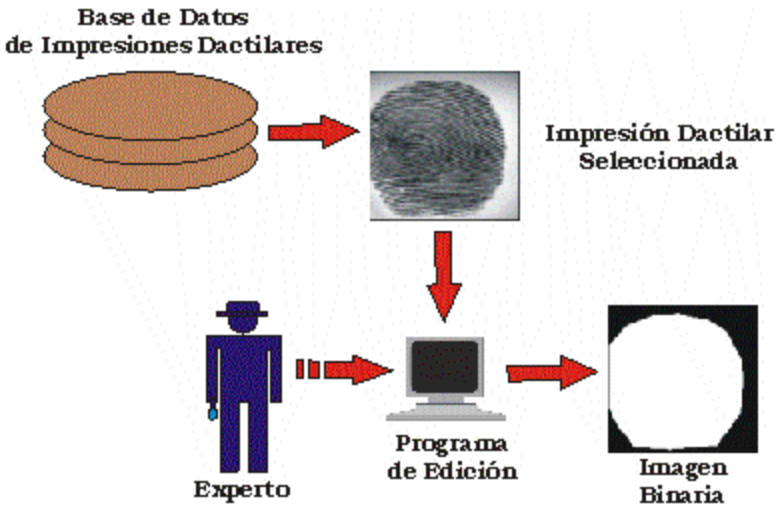


Figura 3.19 Determinación manual de la imagen de regiones reconocibles (blanco) y no reconocibles (negro) de una impresión dactilar realizada por un experto. Imagen 10_5.tif de la base de datos 2 del FVC2002.

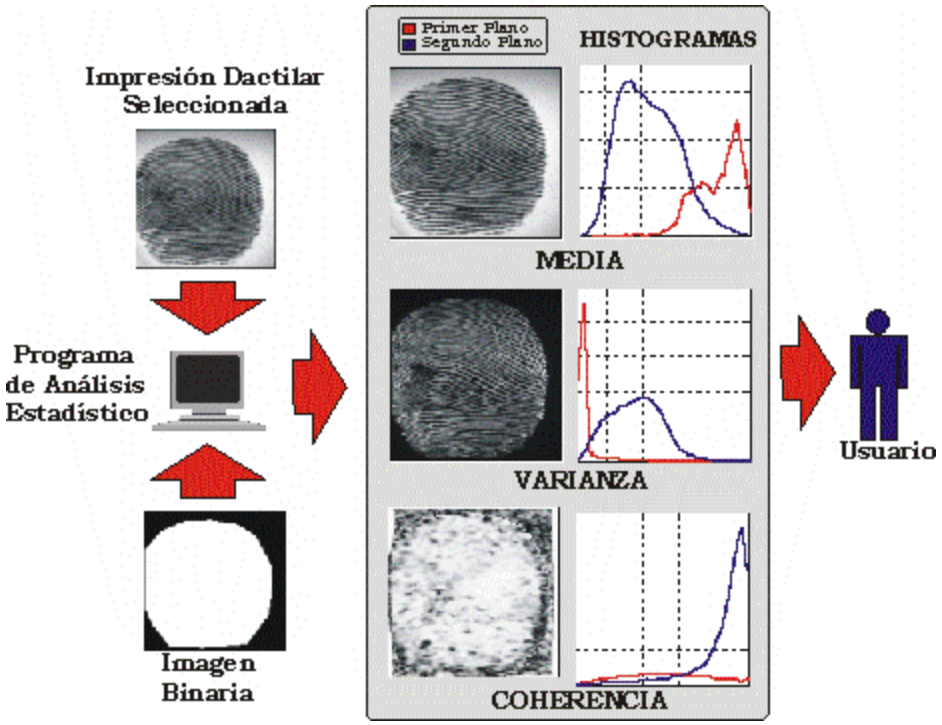


Figura 3.20 Validación manual de la imagen binaria. Imagen 10_5.tif de la base de datos 2 del FVC2002.

3.4.1.1.2. Condensación del conjunto de entrenamiento

El procedimiento de condensado tiene como objetivo redefinir el conjunto de entrenamiento de manera tal que el conjunto resultante esté formado por las muestras que eviten el traslape entre clases en la frontera de decisión. Aquí usamos la técnica de condensado adaptativo basado en los mapas auto organizativos de características (SOFM, ver Anexo D). Esta técnica propuesta inicialmente por [Kohonen, 1982] ha sido utilizada para realizar análisis de agrupamiento de datos (*clustering*) y como se indica en [Cortijo, 1995, pp. 124-125] es útil como herramienta para la condensación del conjunto de entrenamiento.

Luego de obtener los conjuntos de observaciones $\mathbf{X}_{-1}$ y $\mathbf{X}_1$ estos son condensados a través de dos SOFM de manera tal que cada uno condense las observaciones de cada clase, generando los conjuntos de observaciones condensados $\mathbf{Y}_{-1}$ y $\mathbf{Y}_1$ (ver figura 3.21) los cuales conformarán el conjunto de aprendizaje en el paso de clasificación, es decir:

$$\begin{aligned} S_{\mathbf{Y}} &= S_{\mathbf{Y}_{-1}} \cup S_{\mathbf{Y}_1} \\ &= \{(\mathbf{y}_j, c_j) \mid (\mathbf{y}_j \in \mathbf{Y}_{-1} \vee \mathbf{y}_j \in \mathbf{Y}_1) \wedge (c_j \in \Omega)\} \end{aligned} \tag{3.53}$$

Tal que $j = 1,2,..,n_{\mathbf{Y}}$ siendo $n_{\mathbf{Y}}$ la suma del número de observaciones de $S_{\mathbf{Y}_{-1}}$ y $S_{\mathbf{Y}_1}$. La topología de los SOFM queda determinada por el número de características en la que está basada la segmentación y por el número de muestras representativas deseadas por imagen. Para nuestro caso, el número de características es tres y el número de muestras representativas por imagen es de 16x16 (nodos de la capa de competición), esta elección produce una mejora en el proceso de clasificación.

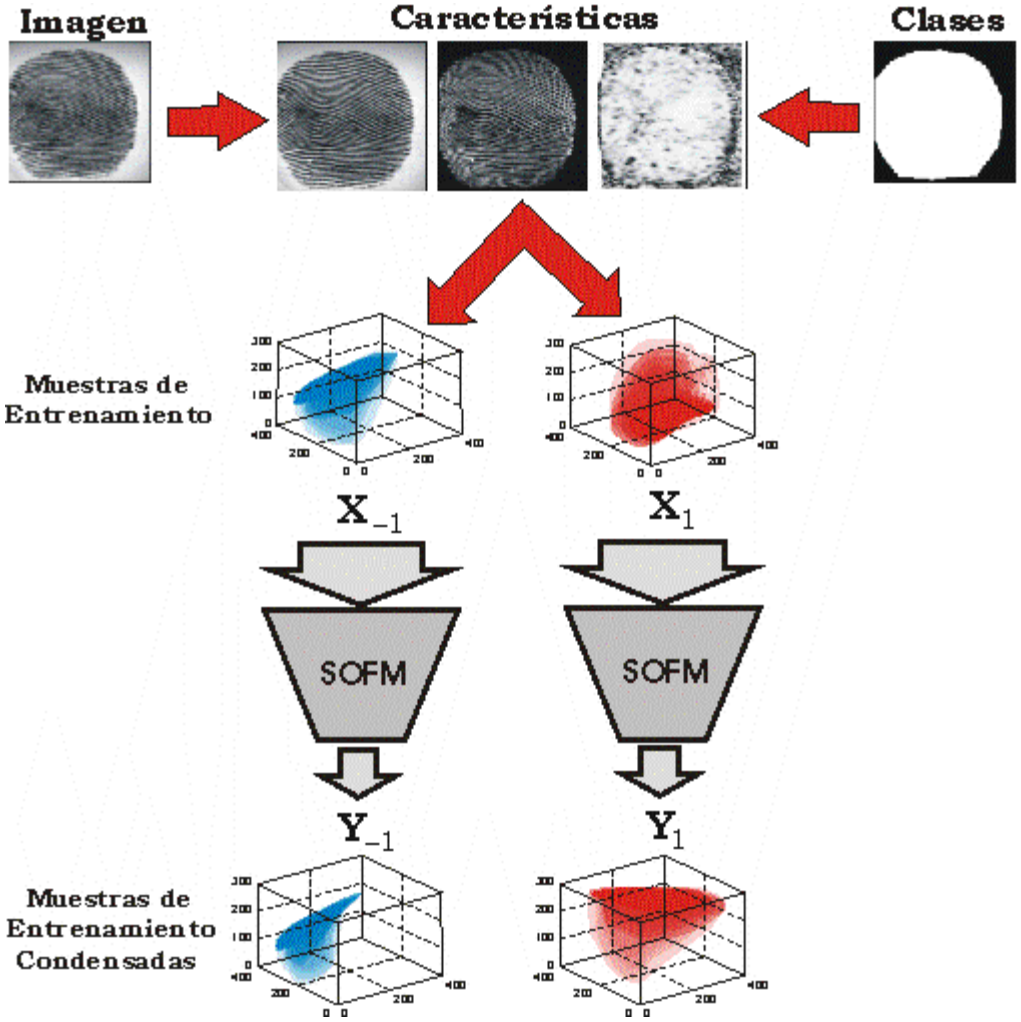


Figura 3.21 Diagrama de los SOFM para cada clase.

3.4.1.1.3. Clasificación

Luego de obtener el conjunto de aprendizaje S_Y se entrena un clasificador para que realice la tarea de segmentación, es decir que clasifique cada píxel de la imagen como perteneciente al primer o segundo plano y como deseamos clasificar todos los píxeles es importante poder usar algoritmos con la menor complejidad computacional posible, tales como los clasificadores lineales supervisados que permiten clasificar espacios linealmente separables.

Hemos seleccionado dos técnicas de entrenamiento para un clasificador lineal: el *Perceptron* y el *Fuzzy Perceptron* (ver Anexo D) ambas mostraron ser computacionalmente adecuadas, sin embargo el *Fuzzy Perceptron* provee una me-

jor tasa de clasificación exitosa debido a su capacidad de discriminación de los datos en la frontera de decisión.

3.4.1.1.4. Post procesamiento

Tal como muestra la figura 3.22 la clasificación de los píxeles tendrá cierto nivel de error como consecuencia de que el *perceptron* es un clasificador lineal, reflejándose en pequeñas áreas espurias de una clase presentes dentro de un área más grande de la otra clase (pequeñas manchas oscuras o bordes no definidos sobre el área definida como primer plano).

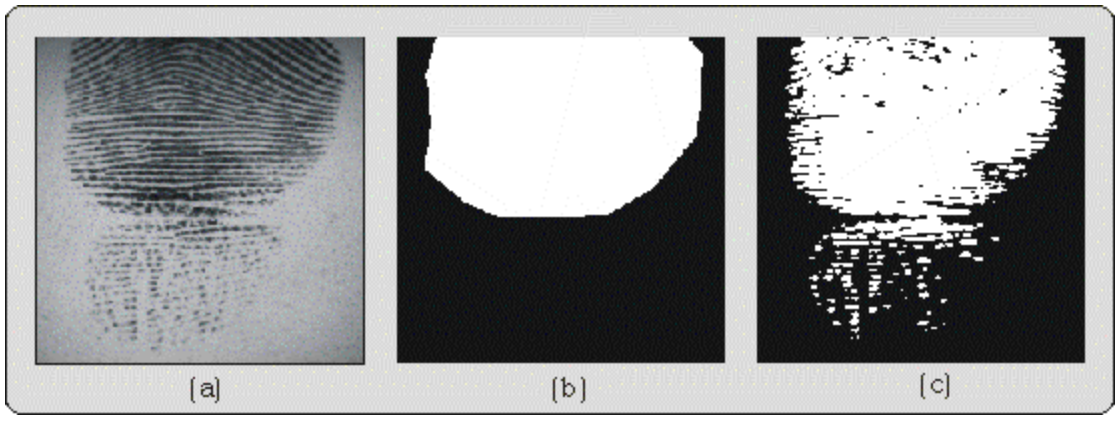


Figura 3.22 (a) Imagen original; (b) imagen de regiones reconocibles determinada por el experto; (c) imagen de regiones reconocibles determinada por el *perceptron*. Imagen 10_4.tif de la base de datos 2 del FVC2002.

Una manera de eliminar las pequeñas regiones mal clasificadas es a través de las técnicas basadas en el procesamiento de imágenes binarias. Definiendo como la imagen binaria a la imagen de regiones reconocibles determinada por el clasificador, donde el primer plano vendrá representado por el objeto binario (color blanco) y el segundo plano vendrá representado por el fondo (color negro). Bajo esta perspectiva la corrección en la clasificación de los píxeles se resume al mejoramiento de los bordes del objeto y la eliminación de los huecos, en [Bazen et al., 2001] se recomienda aplicar morfología matemática para tales tareas. La morfología matemática que comenzó a finales de los años sesenta, forma de alguna manera un cuerpo separado dentro del Análisis de Imágenes, sus principales protagonistas son Matheron y Serra. Es un sistema algebraico de operadores

composicionales que cuando actúan sobre formas complejas son capaces de descomponerlas en sus partes más significativas y separar las partes que le son extrañas. Un sistema de operadores de este tipo y su composición permite que las formas subyacentes sean identificadas y reconstruidas en forma óptima a partir de sus formas distorsionadas y ruidosas. Además permite que cada forma sea entendida en función de una descomposición, siendo cada entidad de esa descomposición una forma simple apropiada. Las operaciones morfológicas pueden simplificar los datos de la imagen, preservar las características esenciales y eliminar aspectos irrelevantes. Dos de tales operadores son la Apertura y el Cierre. La Apertura generalmente suaviza contornos, rompe istmos estrechos y elimina protuberancias delgadas. El Cierre también tiende a suavizar secciones de contornos, pero en oposición la Apertura generalmente fusiona separaciones estrechas, entrantes, delgadas y profundas, eliminando pequeños huecos y rellenando agujeros del contorno. Por lo tanto podemos concluir que:

Los conjuntos de píxeles aislados que son incorrectamente asignados al primer plano son eliminados por medio de una operación de Apertura.

Los conjuntos de píxeles aislados que son incorrectamente asignados al segundo plano son eliminados por una operación de Cierre.

Los resultados obtenidos después de realizar las operaciones de Apertura y Cierre son evidenciados en la figura 3.23, pudiendo apreciarse la eliminación de la mayoría de distorsiones. Debido a la propia naturaleza de la operación de Apertura, ésta podrá generar en la imagen resultante "huecos" (conjunto de píxeles aislados pertenecientes al primer plano inmersos en el segundo plano); debido a ello incluimos una etapa de filtrado binario, eliminando todos aquellos componentes conectados cuya área sea menor que un valor umbral (igual a 300, equivale a un bloque de 30x10 elementos de la imagen) determinado empíricamente.

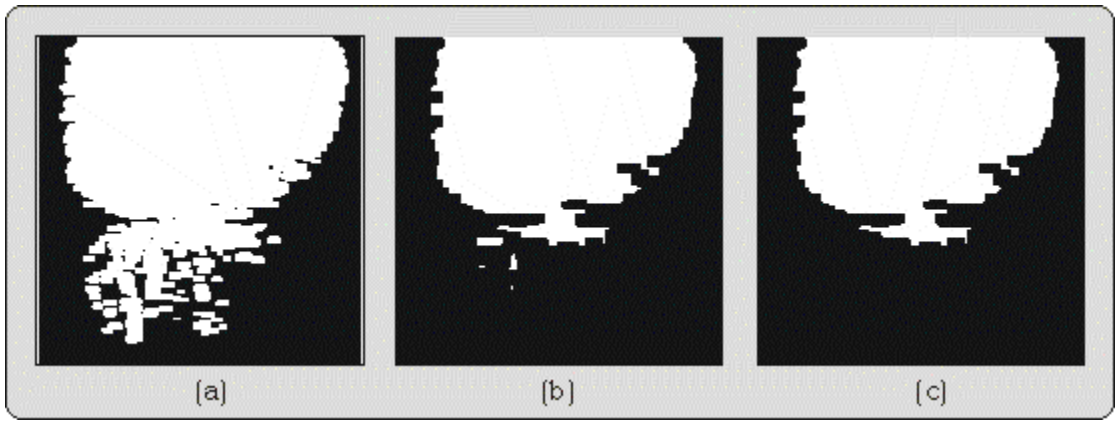


Figura 3.23 (a) Aplicación del operador de Apertura; (b) aplicación del operador de Cierre; (c) eliminación de los huecos.

3.4.1.2. Basada en bloques

El algoritmo implementado para la segmentación de la imagen basada en bloques presenta los siguientes pasos:

1. **Sectorización rectangular de la imagen**: Especifica el número de bloques y el tamaño de cada bloque en los que se va a dividir la imagen.
2. **Extracción de características**: Selecciona las características a nivel de bloques en las cuales se basará la segmentación.
3. **Clasificación**: A través del algoritmo k-NN se obtiene el mapa de regiones deseado, determinándose los centroides del conjunto de entrenamiento.

La sectorización de la imagen es descrita en la sección 3.2.1.1. , a continuación veremos en detalle cada uno de los pasos restantes.

3.4.1.2.2. Extracción de características

Tomando en cuenta los trabajos de [Ratha et al., 1995] y [Maio et al., 1997] seleccionamos dos características básicas: la varianza de los niveles de grises en una vecindad orientada de la imagen y la magnitud promedio del Vector Gradiente. La primera categoriza las regiones considerando la diferencia entre las varianzas de las vecindades orientadas según la orientación local y normal de cada bloque, siendo ésta mayor en el primer plano que en el segundo. La segunda aprovecha que al ser toda impresión dactilar rica en bordes (debido a la alternancia entre

crestas y valles) la magnitud promedio del Vector Gradiente por bloque es alta en el primer plano y baja en el segundo.

Estas características son calculadas para cada bloque (i, j) a través de las expresiones:

$$\mathbf{CG}(i, j) = \frac{1}{N_k{}^2} \sum_{u=(i-1).N_k}^{i.N_k-1} \sum_{v=(j-1).N_k}^{j.N_k-1} \left\|\nabla \mathbf{I}(u, v)\right\|^2 \tag{3.54}$$

$$\mathbf{CV}(i, j) = \frac{1}{N_{rot}} \sum_{u'=0}^{N_{rot}-1} \sum_{v'=0}^{N_{rot}-1} \left\{(\mathbf{I}_{rot}^{(i,j)}(u', v') - Mc_{v'}[\mathbf{I}_{rot}^{(i,j)}])^2 + (\mathbf{I}_{rot}^{(i,j)}(u', v') - Mf_{u'}[\mathbf{I}_{rot}^{(i,j)}])^2\right\} \tag{3.55}$$

Donde:

$\mathbf{CG}(i, j)$: Es la magnitud promedio del Vector Gradiente del bloque (i, j).

$\mathbf{CV}(i, j)$: Es la varianza de una vecindad orientada centrada en el bloque (i, j).

$\left\|\nabla \mathbf{I}(u, v)\right\|$: Es la magnitud del Vector Gradiente en el píxel (u, v).

$\mathbf{I}_{rot}^{(i,j)}(u', v')$: Es la matriz de tamaño $N_{rot} x N_{rot}$ que contiene la vecindad local orientada respecto al bloque $\mathbf{S}_{ij}^k$.

$Mf_u[\bullet]$: Es el operador que calcula el valor promedio de la u-ésima fila de una matriz.

$Mc_v[\bullet]$: Es el operador que calcula el valor promedio de la v-ésima columna de una matriz.

La relación de índices entre el píxel (u, v) del bloque $\mathbf{S}_{ij}^k$ y el elemento (u', v') de la matriz $\mathbf{I}_{rot}^{(i,j)}$ es:

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} (i-1).N_k + \lfloor N_k / 2 \rfloor \\ (j-1).N_k + \lfloor N_k / 2 \rfloor \end{bmatrix} + \begin{bmatrix} \cos(\Theta(i, j)) & -\text{sen}(\Theta(i, j)) \\ \text{sen}(\Theta(i, j)) & \cos(\Theta(i, j)) \end{bmatrix} \begin{bmatrix} u' - \lfloor N_{rot}/2 \rfloor \\ v' - \lfloor N_{rot}/2 \rfloor \end{bmatrix} \tag{3.56}$$

Cumpliéndose la igualdad (ver figura 3.24):

$$\mathbf{I}_{rot}^{(i,j)}(u',v') = \mathbf{I}(u,v) \text{ , para todo } 0 \le u', v' \le N_{rot} - 1 \tag{3.57}$$

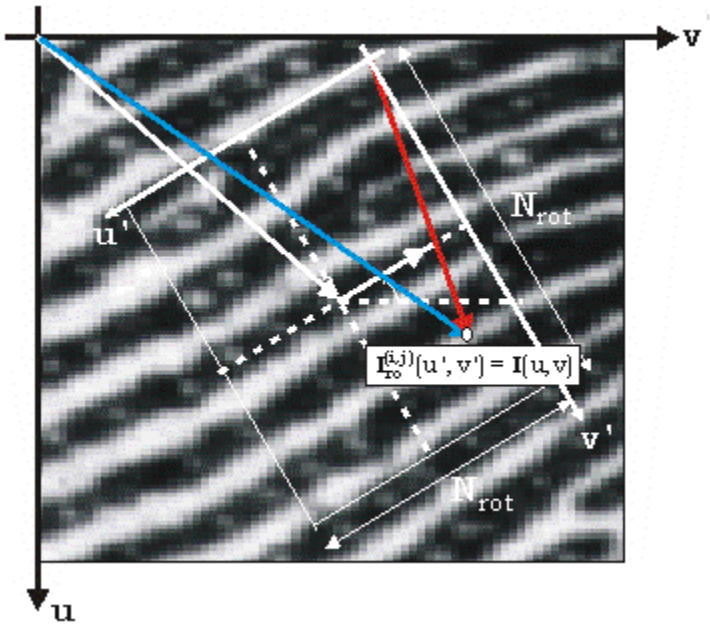


Figura 3.24 Relación de los ejes coordenados vinculados a la imagen (u,v) y a la vecindad orientada (u',v').

Estas dos características definen la observación asociada a cada bloque $\mathbf{S}_{ij}^{k}$, es decir:

$$\mathbf{x} = [\mathbf{CG}(i,j) \quad \mathbf{CV}(i,j)]^T ; \mathbf{x} \in \square^{\mathbf{2}} \tag{3.58}$$

En la figura 3.25 se muestran las imágenes de **CG** y **CV** en donde podemos apreciar que ambas permiten diferenciar fácilmente entre el primer y segundo plano.

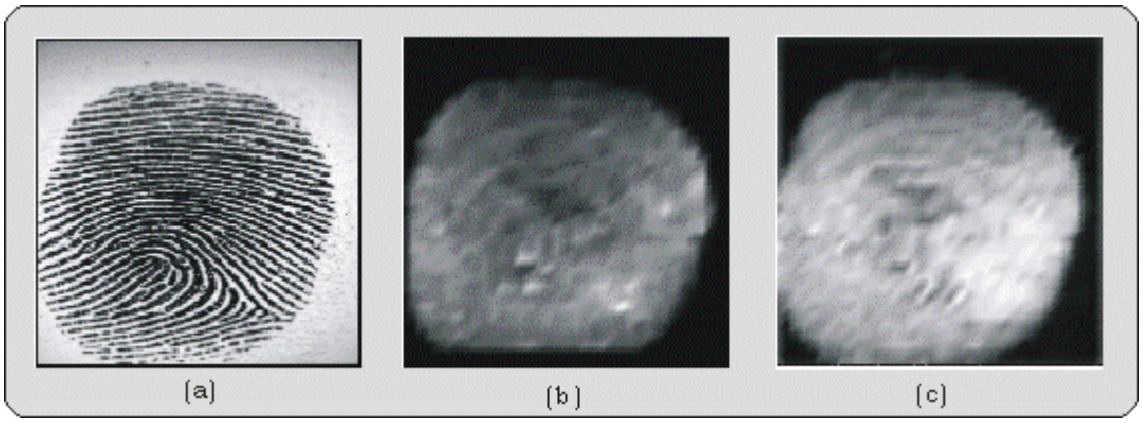


Figura 3.25 Características extraídas basadas en bloques para la segmentación (a) imagen original; (a) varianza de una vecindad orientada; (b) magnitud promedio del Vector Gradiente. Imagen 13_6.tif de la base de datos 2 del FVC2002.

3.4.1.2.3. Clasificación

Luego de obtener la observación asociada a cada bloque utilizamos un clasificador k-NN que realizará la tarea de segmentación, clasificando cada bloque $\mathbf{S}_{ij}^{k}$ como perteneciente al primer o segundo plano (ver figura 3.26). El clasificador k-NN es ampliamente usado debido a su alta precisión y simplicidad. La idea básica detrás del k-NN es la ubicación de sus vecinos cercanos. Dada la observación $\mathbf{x}$ a clasificar como perteneciente al primer o segundo plano, la regla de clasificación k-NN (para nuestro caso k es igual a dos) está basada en determinar la distancia Euclidiana entre la observación $\mathbf{x}$ y los centroides representativos de cada clase $\mathbf{x}_{-1}^{ctr}$ y $\mathbf{x}_{1}^{ctr}$, asignando como la clase más probable a aquella cuyo centroide se encuentre a menor distancia de dicha observación. Para la determinación de los centroides $\mathbf{x}_{-1}^{ctr}$ y $\mathbf{x}_{1}^{ctr}$ vinculados a las clases w_{-1} y w_{1}, seguimos los siguientes pasos:

1. Para las imágenes seleccionadas siguiendo las pautas establecidas en la sección 3.4.1.1, determinamos las observaciones a nivel de bloques, obteniéndose las matrices respectivas $\mathbf{X}_{-1}$ y $\mathbf{X}_{1}$.

2. Para cada matriz determinamos el promedio estadístico de todas sus observaciones de entrenamiento, obteniéndose los centroides representativos de las clases en cuestión.

$$\mathbf{x}_{-1}^{ctr} = \frac{1}{n_{-1}} \sum_{j=1}^{n_{-1}} \mathbf{X}_{j:-1} = [\mu_{CG}^{-1} \quad \mu_{CV}^{-1}]^T \tag{3.59}$$

$$\mathbf{x}_{1}^{ctr} = \frac{1}{n_{1}} \sum_{j=1}^{n_{1}} \mathbf{X}_{j:1} = [\mu_{CG}^{1} \quad \mu_{CV}^{1}]^T \tag{3.60}$$

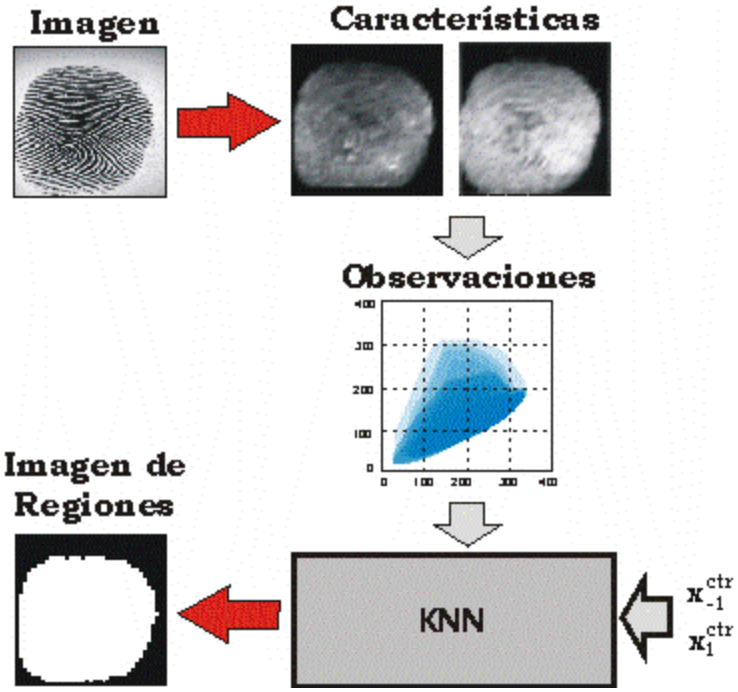


Figura 3.26 Esquema para la clasificación de los bloques $\mathbf{S}_{ij}^{k}$ basada en el k-NN.

3.5. Resultados Experimentales

3.5.1. En la estimación local de la orientación

Los resultados obtenidos en esta sección reflejan el comportamiento de los algoritmos de promediación del Vector Gradiente por vecindades ante la presencia del ruido, para lo cual seguimos las siguientes pautas:

1. Se añade a la imagen de prueba una señal de ruido Gaussiano con una desviación estándar específica, obteniéndose una imagen ruidosa.

2. Se aplican los algoritmos de promediación del Vector Gradiente, tanto a la imagen de prueba como a la imagen ruidosa.

3. Se calculan y almacenan los errores producidos por los resultados correspondientes a la imagen de prueba y a la imagen ruidosa.

4. Se repiten los pasos del 1 al 3 pero con diferente desviación estándar, como resultado se obtiene una curva de error que nos permite discernir acerca de que algoritmo es el más eficiente respecto al ruido.

Si no se incluye la etapa de filtrado en ambas técnicas de promediación, éstas presentan el mismo comportamiento, es decir tienen los mismos valores para diferentes niveles de ruido, por lo que será indiferente usar uno u otro. Pero al incluir el filtrado, las curvas de error muestran una apreciable diferencia en el desempeño. La figura 3.27 muestra la curva de error obtenida al aplicar la promediación del Vector Gradiente y puede observarse que la mejora no es muy significativa, sin embargo la figura 3.28 muestra la curva de error obtenida al aplicar el cálculo de la SVD, la robustez respecto al ruido es apreciable para valores menores al 40% de la máxima desviación estándar permitida. Por lo tanto, es obvio que la SVD incluyendo la etapa de filtrado es superior ante la presencia de ruido Gaussiano, esto queda validado en la figura 3.29 que muestra la curva de error para ambas técnicas. Sin embargo, tomando en cuenta que la SVD tiene un costo computacional mayor que la basada en la promediación del Vector Gradiente, es recomendable utilizar la primera sólo para aplicaciones de adquisición *off-line* o en imágenes muy ruidosas.

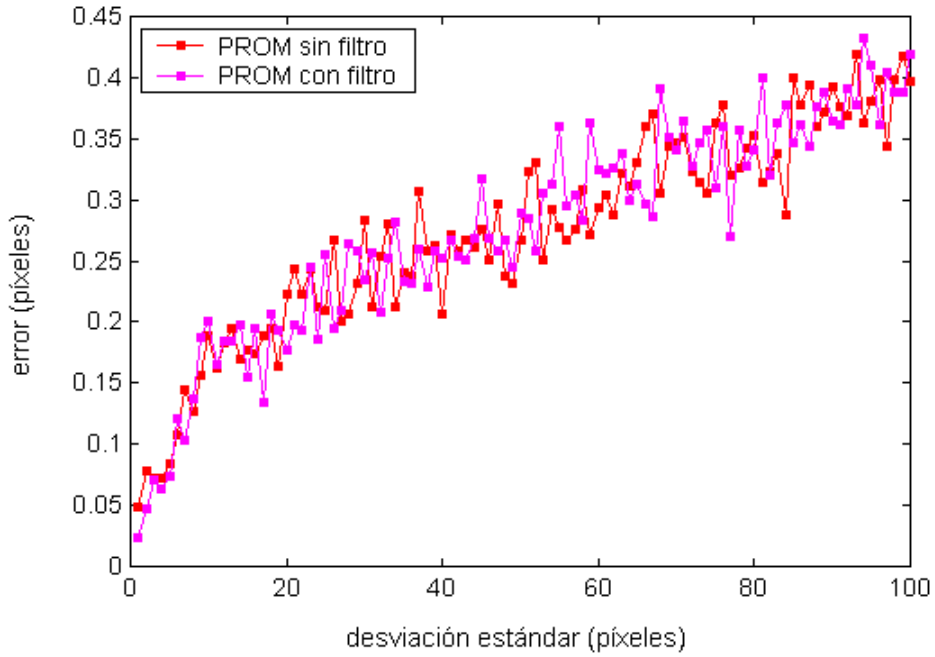


Figura 3.27 Curva de error para la técnica de promediación del Vector Gradiente con y sin la etapa de filtrado.

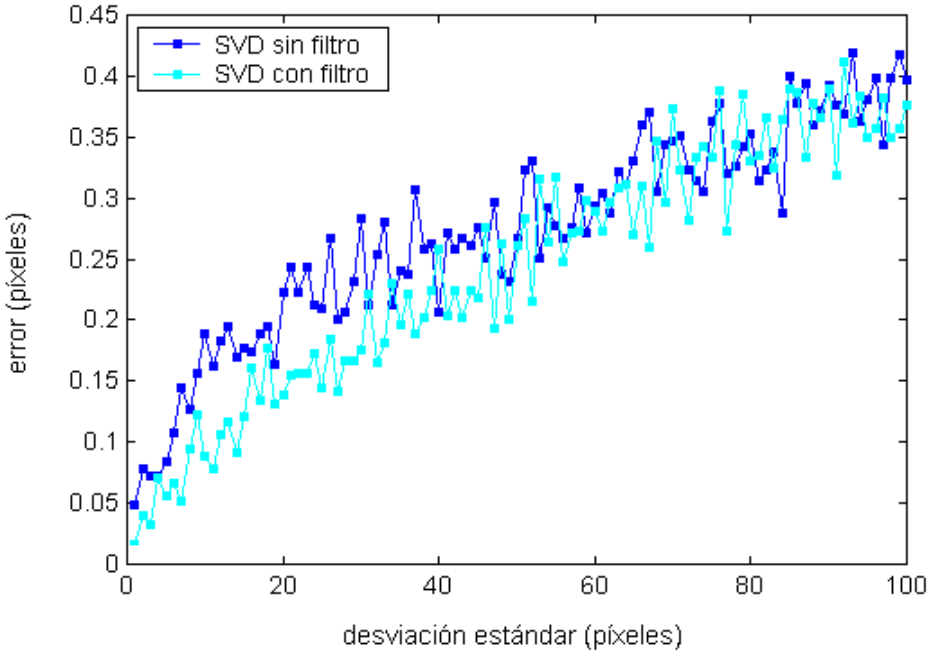


Figura 3.28 Curva de error para la técnica basada en la SVD con y sin la etapa de filtrado.

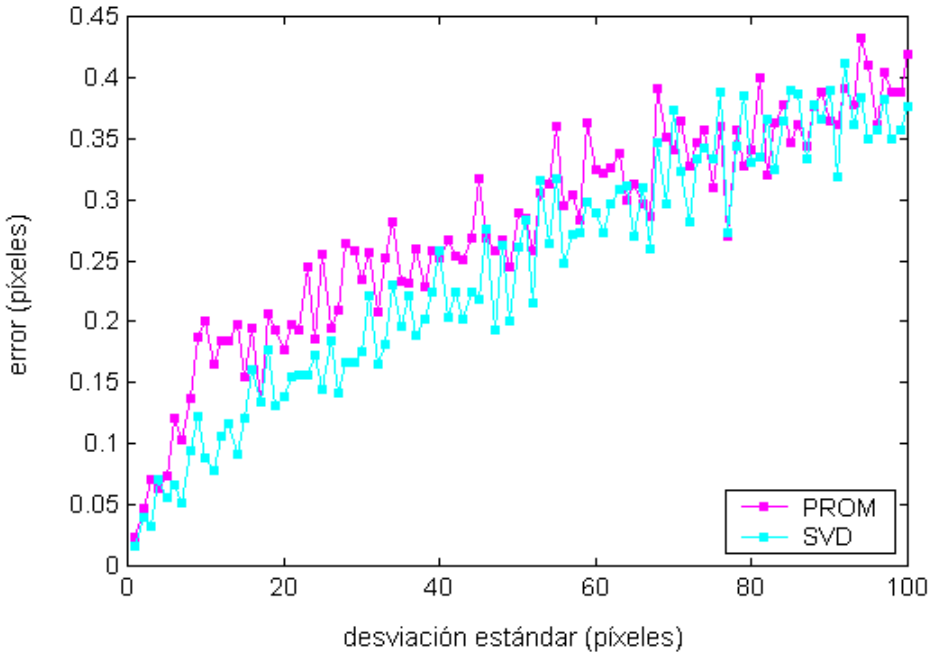


Figura 3.29 Comparación de las curvas de error para ambas técnicas incluyendo la etapa de filtrado.

3.5.2. En la estimación de la Imagen de Frecuencias

Para evaluar el algoritmo de estimación de la Imagen de Frecuencias fue necesario contar con una imagen que presente una frecuencia espacial conocida en todas sus posibles vecindades, por lo cual se desarrolló una función que permite generar superficies sinusoidales orientadas con frecuencia espacial específica. En la figura 3.30 se aprecian dos superficies sinusoidales orientadas a 30º y con frecuencias espaciales de 1/10 y 1/20 píxeles^{-1} respectivamente.

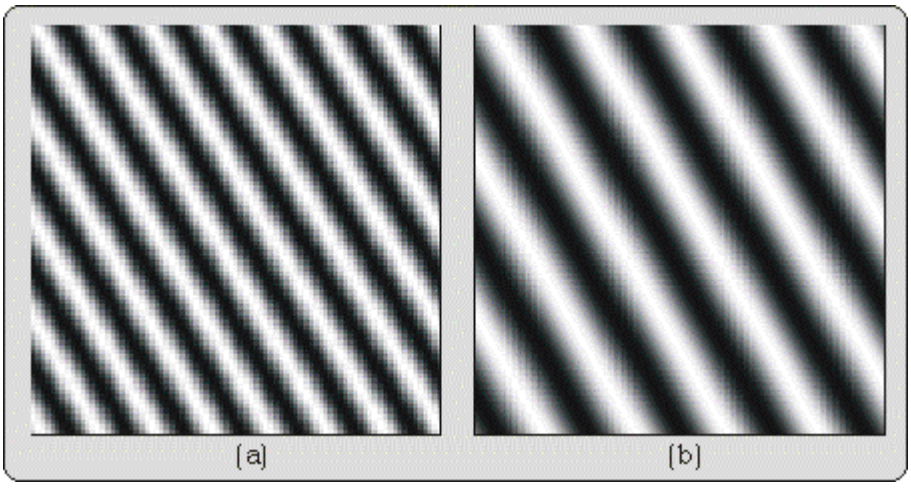


Figura 3.30 Imágenes de superficies sinusoidales orientadas, con frecuencias espaciales de (a) 1/10 píxel^{-1}; (b) 1/20 píxel^{-1}.

Se han realizado dos pruebas sobre estas imágenes, la primera para establecer el intervalo sobre el cual el tamaño de la vecindad genera un mínimo error en la estimación de la frecuencia espacial y la segunda para determinar el efecto del ruido en esta estimación.

Los resultados de la primera prueba son presentados en la figura 3.31, en donde se observa que para el intervalo de frecuencias espaciales <1/20,1/2> píxeles^{-1} cualquier vecindad que esté entre 15 y 35 píxeles por lado genera un mínimo error, para intervalos de frecuencia superiores el error es creciente. Puesto que la mayoría de las impresiones dactilares capturadas a 500 dpi tienen una frecuencia espacial en el intervalo <1/15,1/5> píxeles^{-1} sería adecuado escoger un tamaño de ventana de 25 (=(15+35)/2) píxeles por lado lo que nos asegura un mínimo error en la estimación.

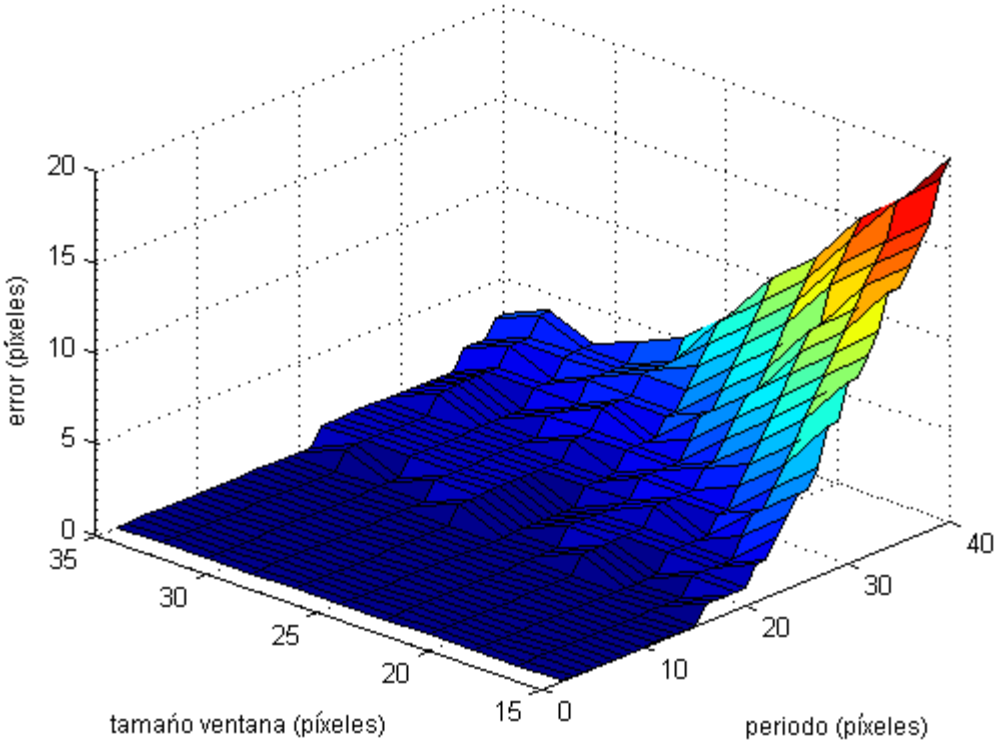


Figura 3.31 Gráfica del error en la estimación de la frecuencia espacial respecto al tamaño de la ventana.

Los resultados de la segunda prueba son presentados en la figura 3.32, obtenida usando una vecindad de 25 píxeles por lado (por los motivos expuestos en el párrafo anterior) y un ruido aditivo Gaussiano caracterizado por su desviación estándar. En esta figura se puede apreciar que el algoritmo es relativamente inmune al ruido (excepto para períodos altos) cuando presenta una desviación estándar en el intervalo <0,10> píxeles, sin embargo no es concluyente que esta inmunidad se mantenga para toda impresión dactilar, puesto que la naturaleza del ruido que las caracteriza no necesariamente viene representado por ruido aditivo o por la suma de ruidos aditivos, sólo nos permite concluir que el algoritmo calcula con éxito la frecuencia espacial por vecindad de la imagen aunque ésta sufra grandes variaciones en sus niveles de grises.

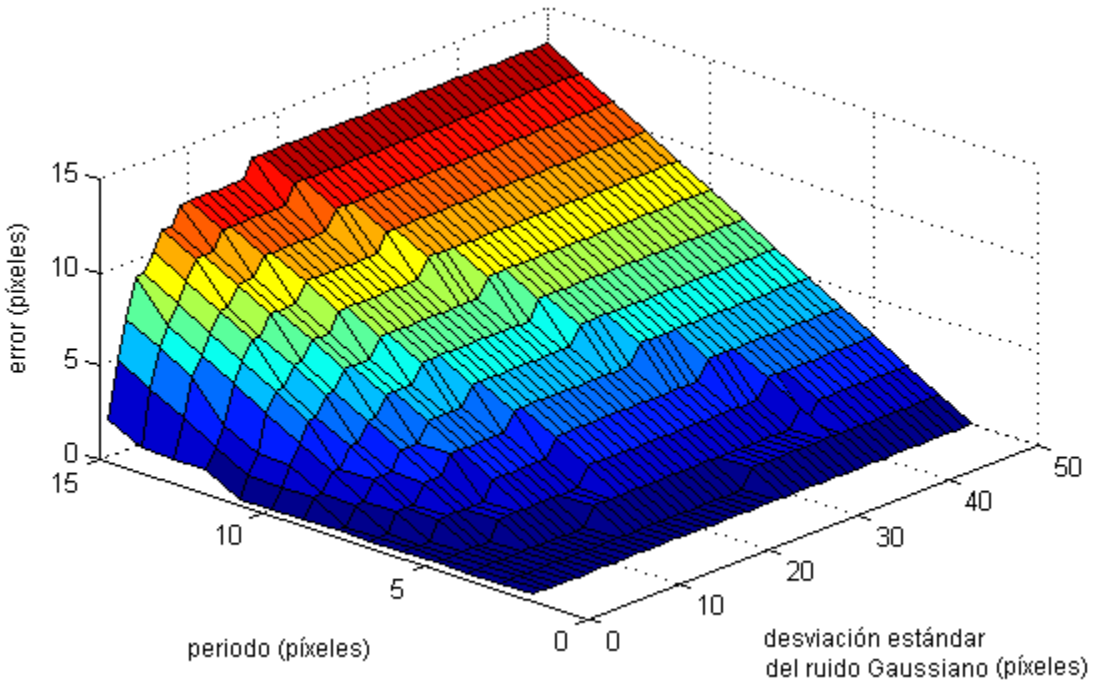


Figura 3.32 Gráfica del error en la estimación de la frecuencia espacial respecto a la desviación estándar del ruido Gaussiano.

3.5.3. En la estimación de la Imagen de Regiones

3.5.3.1. Obtención de las muestras de entrenamiento

En este experimento utilizamos la base de datos 2 del FVC2002, de donde seleccionamos las imágenes: 10_3.tif, 10_4.tif, 10_5.tif, 11_7.tif, 12_7.tif, 13_6.tif, 14_2.tif, 14_6.tif, 18_4.tif, 18_6.tif, 19_6.tif, 22_1.tif, 22_6.tif, 32_3.tif, 36_2.tif, 40_1.tif, 42_3.tif, 52_8.tif, 53_2.tif, 55_7.tif,59_2.tif, 60_2.tif, 63_6.tif, 64_5.tif, 66_6.tif, 67_7.tif, 72_6.tif, 73_6.tif, 76_2.tif, 84_5.tif, 95_6.tif, 99_1.tif, 100_3.tif, 100_6.tif. Para cada imagen siguiendo el procedimiento explicado en la sección 3.4.1.1. determinamos manualmente la imagen de regiones reconocibles.

Para la clasificación a nivel de píxeles, puesto que son 34 imágenes de tamaño 560x296 píxeles se generan 5 635 840 (=34x560x296) muestras de entrenamiento, que conforman el *corpus* inicial. Para eliminar las muestras redundantes, condensamos este *corpus* tal como fue explicado en la sección 3.4.1.1. a través de dos SOFM de 3 neuronas en la capa de entrada y 16x16 neuronas en la capa de

competición, obteniéndose un total de 8 704 observaciones por cada clase. En la figura 3.33 mostramos los histogramas vinculados a cada característica, tanto para las muestras sin condensar (a) como condensadas (b), puede apreciarse que después de la condensación el traslapamiento entre clases es menor y los histogramas tienen una apariencia suavizada puesto que únicamente quedan las muestras de entrenamiento más representativas.

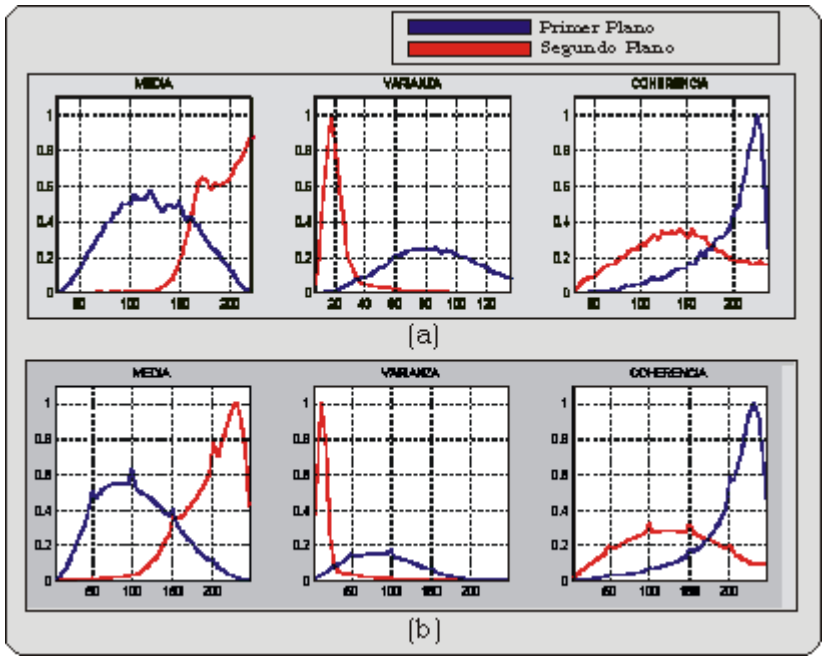


Figura 3.33 Histogramas para cada característica (a) antes de la condensación; (b) después de la condensación.

Para la clasificación a nivel de bloques tomamos las mismas imágenes de entrenamiento que en el caso anterior, sectorizándolas en bloques de tamaño 8x4 píxeles, generándose 176 120 (=34x(560/8)x(296/4)) muestras de entrenamiento. Promediando las muestras pertenecientes al primer y segundo plano se obtienen los centroides $\mathbf{x}_{-1}^{ctr} = [14 \quad 24]^T$ y $\mathbf{x}_{1}^{ctr} = [76 \quad 105]^T$ utilizados para la clasificación a través del k-NN. En la figura 3.34 se aprecian las nubes de puntos formadas por las muestras de entrenamiento pertenecientes al primer y segundo plano así como sus respectivos centroides. Se debe tomar en cuenta que para este caso no fue necesario condensar las muestras, puesto que se desea obtener una medida de promediación requiriéndose la mayoría de los datos.

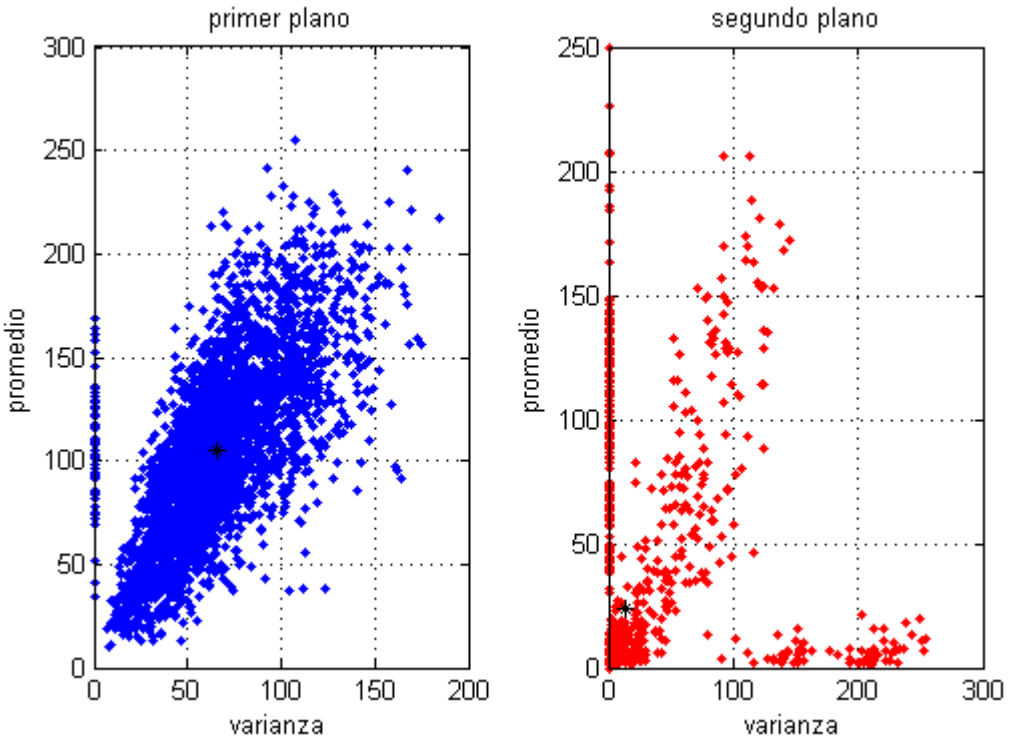


Figura 3.34 Nubes de puntos vinculados a $\mathbf{X}_{-1}$, $\mathbf{X}_{1}$ y sus respectivos centroides.

3.5.3.2. Clasificación

Para la clasificación a nivel de píxeles se implementaron las dos técnicas de aprendizaje para entrenar al clasificador lineal, descritas en el Anexo D. Teniendo como conjunto de entrenamiento a $\mathbf{S_Y}$ se obtuvo una considerable mejora en el tiempo de entrenamiento para la convergencia del algoritmo del *Perceptron*, pues comparado con las 10^6 épocas que fueron reportadas por [Bazen et al., 2001], se logró una reducción al orden de 10^4 en ambas técnicas de aprendizaje. Para reducir la probabilidad de que los entrenamientos *Perceptron* y *Fuzzy Perceptron* queden estancados en un mínimo local se utilizó un protocolo de entrenamiento por lotes, es decir se presentaron todas las entradas con sus respectivas salidas, calculándose los errores parciales para todo el *corpus* de entrenamiento, luego se promediaron para obtener un error global, actualizándose el vector de pesos únicamente a través de este error, repitiéndose estos pasos hasta que se cumpla con el criterio de terminación. Además para el entrenamiento *Fuzzy Perceptron* se obtuvieron las constantes f y ξ a través del método de prueba y error (ver sección

D.5.3.), puesto que ambas permiten al usuario definir el grado de influencia positiva o negativa que poseen las muestras cercanas a la frontera de decisión. Los resultados muestran que en un menor número de épocas el entrenamiento *Fuzzy Perceptron* provee un hiperplano de separación más óptimo que el obtenido por el entrenamiento del *Perceptron*, disminuyendo su error promedio en aproximadamente 1.5%. La figura 3.35 muestra una comparación de los resultados obtenidos utilizando las técnicas de entrenamiento *Perceptron* y *Fuzzy Perceptron*.

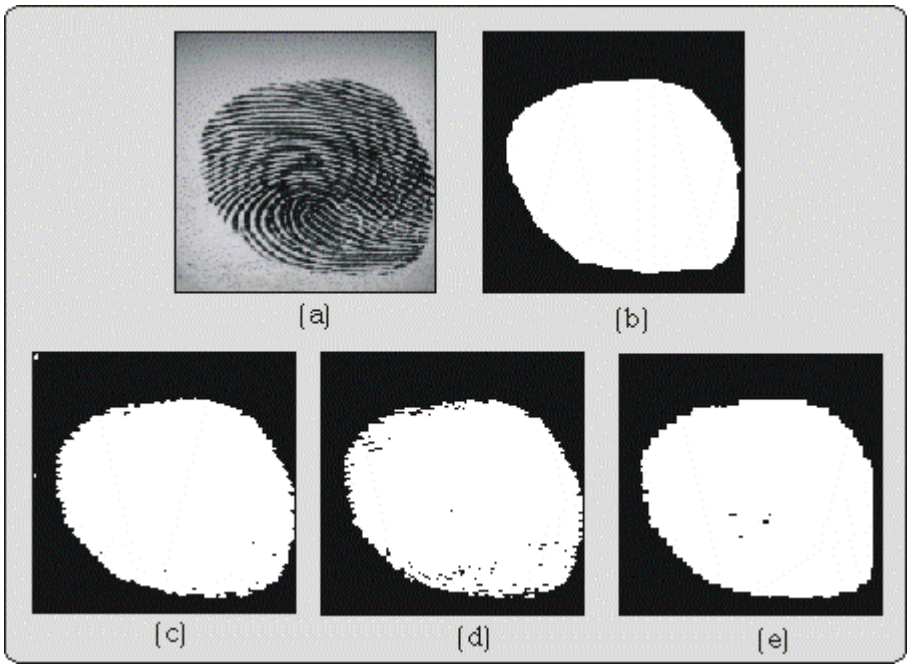
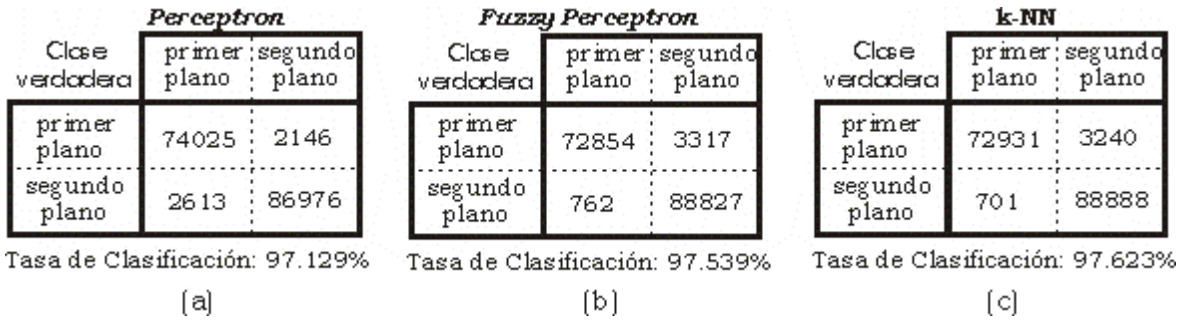


Figura 3.35 Resultados obtenidos por los clasificadores a nivel de píxeles y bloques (a) imagen original; Imágenes de Regiones: (b) obtenida manualmente; (c) obtenida por el entrenamiento *Fuzzy Perceptron;* (d) obtenida por el entrenamiento *Perceptron;* (e) obtenida por el k-NN.

Para la clasificación a nivel de bloques se utilizó un clasificador k-NN para k=2 teniendo como métrica la distancia Euclidiana y como centroides los obtenidos en la sección anterior. La gran ventaja respecto a los trabajos de [Ratha et al., 1995] y [Maio y Maltoni, 1997] es que no requiere de valores umbrales, puesto que nos valemos de las muestras de entrenamiento anteriores para obtener los parámetros del k-NN.



Perceptron

Clase verdadera	primer plano	segundo plano
primer plano	74025	2146
segundo plano	2613	86976

Tasa de Clasificación: 97.129%

(a)

Fuzzy Perceptron

Clase verdadera	primer plano	segundo plano
primer plano	72854	3317
segundo plano	762	88827

Tasa de Clasificación: 97.539%

(b)

k-NN

Clase verdadera	primer plano	segundo plano
primer plano	72931	3240
segundo plano	701	88888

Tasa de Clasificación: 97.623%

(c)

Figura 3.36 Matrices de Confusión: (a) para el *Perceptron;* (b) para el *Fuzzy Perceptron;* (c) para el k-NN.

La figura 3.36, muestra los resultados experimentales obtenidos a través de la Matriz de Confusión [Aching y Rojas, 2003] del *Perceptron* y del *Fuzzy Perceptron* (a nivel de píxeles) y del k-NN (a nivel de bloques) para la imagen 60_2.tif.

3.6. Resumen

Hemos analizado, implementado y comparado dos de los algoritmos más sofisticados para la estimación de la Imagen de Orientación, éstos algoritmos nos permiten caracterizar y cuantificar a nivel local la propiedad de orientación de las crestas en la imagen completa, suministrando a su vez información acerca de las características topológicas de la impresión dactilar, la que es utilizada principalmente como un mecanismo de indexación en grandes bases de datos. Ambos algoritmos poseen tanto ventajas, así como también desventajas, sin embargo una adecuada elección depende fuertemente de la aplicación, puesto que la principal diferencia radica en los requerimientos computacionales y en la robustez frente al ruido.

Se ha implementado un algoritmo eficiente y robusto para la estimación de la Imagen de Frecuencias propuesto en la literatura, que provee de información fundamental para tareas muy importantes en la extracción de características, tales como el mejoramiento de la impresión dactilar, así como también para la detección de minucias directa en escala de grises (ver [Maio y Maltoni, 1997]). La principal característica de este algoritmo es su baja complejidad computacional, haciéndolo adecuado para aplicaciones que requieren un corto tiempo de procesamiento.

Además se han implementado y propuesto mejoras en las dos técnicas planteadas en la literatura para la estimación de la Imagen de Regiones: a nivel de píxeles (supervisada) y a nivel de bloques (no supervisada). En la segmentación por píxeles, se ha propuesto un procedimiento para obtener un conjunto de entrenamiento adecuado utilizando como entrada tres características de los píxeles, siendo éstas condensadas a través de dos SOFM para obtener un sub conjunto representativo óptimo que es utilizado para la clasificación de las regiones. En la segmentación por bloques, se selecciona un conjunto de entrenamiento de manera similar al desarrollado en la segmentación por píxeles, pero con dos características diferentes y calculadas a nivel de bloques, además en vez de seleccionar algún valor umbral empírico vinculado a cada característica para discriminar entre regiones (como es usualmente realizado), se determinaron los centroides de cada clase conformantes del conjunto de entrenamiento inicial, utilizándolos como parámetros de entrada para un clasificador k-NN.

En la segmentación por píxeles, la condensación del conjunto de entrenamiento es un paso muy importante, pues aumenta la probabilidad de convergencia del algoritmo de entrenamiento del clasificador lineal en un número reducido de épocas. Así mismo, el aprendizaje *Fuzzy Perceptron* provee una forma robusta y eficiente de encontrar un hiperplano de separación lineal, sobretodo cuando el problema posee muestras de entrenamiento no lineales y con traslape entre clases, puesto que permite definir el grado de influencia que tendrán las muestras con mayor incertidumbre o cercanas a la frontera de decisión.

Los algoritmos de segmentación por bloques propuestos en la literatura presentan baja complejidad computacional y los resultados obtenidos son bastante aceptables, sin embargo están fuertemente influenciados por el ruido presente y son dependientes de ciertos valores umbrales empíricos que pueden variar de acuerdo a ciertas características de la imagen y del ruido. Por ello hemos propuesto utilizar los centroides de un conjunto de entrenamiento para hacer más robusto el algoritmo y lo menos sensible al ruido.

Capítulo IV
Extracción de Características de la Impresión Dactilar

4.1. Introducción

¿Qué información está disponible en las impresiones dactilares para dar un juicio válido al momento de decidir si dos impresiones tienen un origen común?. Establecer fiablemente este hecho, requiere necesariamente capturar alguna característica de la huella dactilar que permanezca relativamente inalterada al paso del tiempo, a los cortes, a las contusiones y a las distorsiones introducidas por el dispositivo de adquisición. Como se presenta en la sección 2.3.1. las características que cumplen con estos requisitos pueden categorizarse en dos tipos: globales y locales. Las características globales son atributos válidos para la impresión dactilar entera, mientras que las características locales representadas por las minucias son las que definen exclusivamente la singularidad de la impresión dactilar. Usualmente, se emplean las características globales para el proceso de clasificación y las características locales para el proceso de coincidencia. La extracción de las características tanto globales como locales requiere de técnicas basadas en el procesamiento de imágenes, teoría estadística y reconocimiento de patrones, por lo que el proceso de extracción es uno de los pasos de mayor complejidad e importancia, además una óptima detección de las características genera un alto rendimiento en el sistema de reconocimiento.

4.2. Extracción de las características globales

Las características globales más significativas permiten catalogar a las impresiones dactilares dentro de un conjunto de clases o tipos, existiendo varios esquemas de clasificación, donde uno de los más simples clasifica las impresiones dactilares dentro de las siguientes seis principales clases: *Arch* (A), *Tented arch* (TA), *Left loop* (LL), *Right loop* (RL), *Twin loop* (TL), *Whorl* (W). Algunas veces, una

categoría alterna llamada *Scars* (S) se incluye para la clasificación de huellas dactilares mutiladas con cicatrices que evitan la posibilidad de determinar con precisión su clase verdadera. La determinación de la clase a la que pertenece la impresión dactilar es obtenida mediante el conocimiento de sus puntos singulares (*Core* y *Delta*). Otra característica global de importancia viene a ser el punto de referencia, puesto que en la mayoría de sistemas de verificación e identificación por huellas dactilares se requiere determinar un punto de referencia para lograr una extracción de características robusta respecto a la rotación y traslación. Definimos el punto de referencia de una impresión dactilar como el punto de máxima curvatura de la concavidad de las crestas (ver figura 4.1).

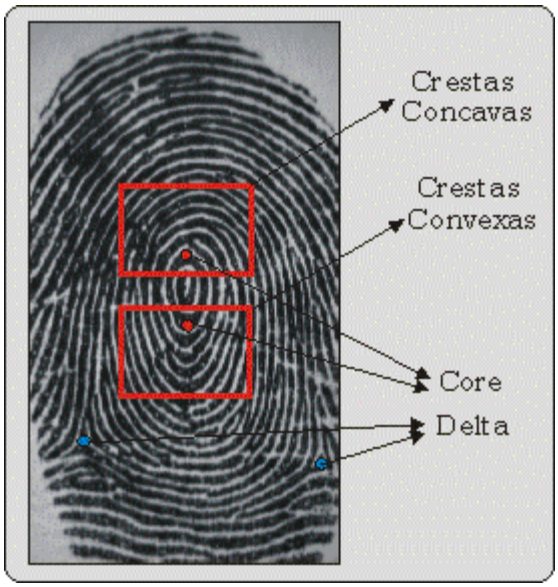


Figura 4.1 Puntos singulares y sectores pertenecientes a las crestas cóncavas y convexas en una impresión dactilar. Imagen 18_6.tif de la base de datos 2 del FVC2002.

4.2.1. Extracción de los puntos singulares de la impresión dactilar

La mayoría de técnicas propuestas en la literatura para la extracción de los puntos singulares operan sobre la Imagen de Orientación, definiéndose los puntos singulares como discontinuidades presentes en ésta. En este trabajo implementamos

dos técnicas que utilizan la Imagen de Orientación como fuente de información. La primera comúnmente empleada por su sencillez es la técnica basada en el índice de Poincaré propuesta por [Kawagoe y Tojo, 1984] que incluye las mejoras recomendadas en [Karu y Jain, 1996] y [Zhang et al., 2001]. La segunda es una técnica novedosa propuesta en [Ramö et al., 2001] que permite determinar las singularidades de la Imagen de Orientación a través de la utilización de sus componentes sinusoidales. La determinación de la Imagen de Orientación fue explicada en detalle en la sección 3.2.1. Recordemos algunos aspectos de mayor importancia, éste se denota por Θ y representa las orientaciones locales de las crestas, no se calcula para cada píxel de la imagen sino para una determinada vecindad, por lo que es una matriz de menor tamaño que la imagen original.

4.2.1.1. Técnica basada en el índice de Poincaré

Si $\mathbf{C}$ es una curva cerrada inmersa en Θ, el índice de Poincaré $\mathbf{P}_{\Theta,\mathbf{C}}$ se define como la rotación total de las líneas de flujo de Θ a lo largo de $\mathbf{C}$ (ver figura 4.2), es decir es la suma de los cambios en la dirección angular alrededor de la curva. El índice de Poincaré $\mathbf{P}_{\Theta,\mathbf{C}}$ para cada posición (i, j) de la Imagen de Orientación se obtiene como sigue:

1. Definimos a $\mathbf{C}_r$ como una curva cerrada que establece una secuencia ordenada de algunos elementos de Θ, tal que (i, j) es un punto interno y el subíndice r denota el orden de la curva, para nuestro caso $\mathbf{C}_r$ será el contorno de un cuadrado de $N_{\mathbf{C}_r} x N_{\mathbf{C}_r}$ píxeles, tal que: $N_{\mathbf{C}_r} = (2r + 1)$ para r mayor que cero, si r es igual a cero $\mathbf{C}_r$ será el contorno de un cuadrado 2x2 bloques. dicho contorno es recorrido en dirección contraria a las agujas del reloj, tal como se indica en la figura 4.2.

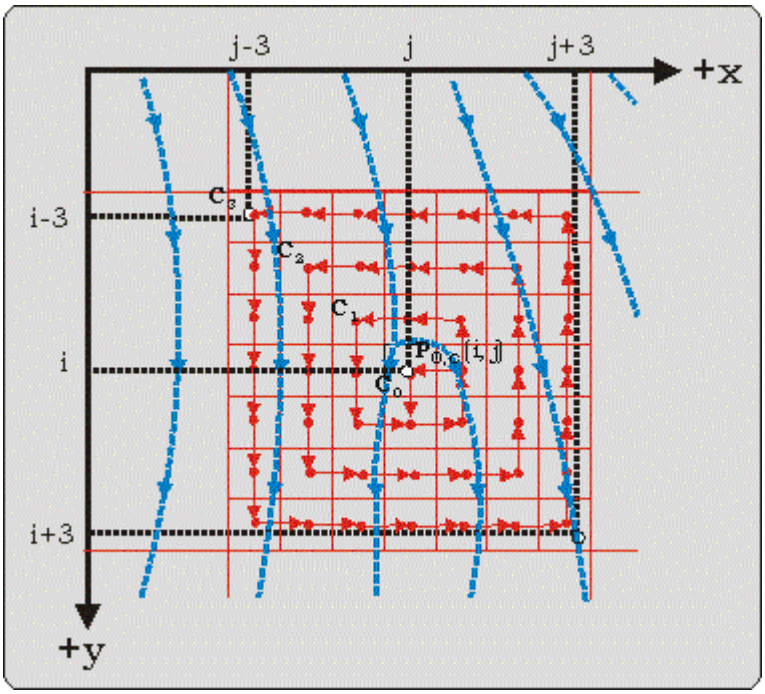


Figura 4.2 Curvas cerradas para la obtención del índice de Poincaré.

El tamaño de $\mathbf{C}_r$ es crucial para el buen rendimiento del algoritmo. Si es muy pequeño, entonces una mínima perturbación en la orientación puede resultar en la detección de puntos singulares espurios. Por otro lado, si es muy grande los puntos singulares verdaderos pueden ser ignorados. Una manera de atenuar la detección de puntos singulares espurios (como se recomienda en [Karu y Jain, 1996]) es suavizar la Imagen de Orientación, de manera tal que los puntos singulares de mayor relevancia sean los únicos que perduren.

2. $\mathbf{P}_{\Theta,\mathbf{C}}(i, j)$ es calculado como la suma algebraica de las diferencias en las orientaciones entre los elementos adyacentes de la curva $\mathbf{C}_r$. Si consideramos a $\varepsilon_X(k)$ y $\varepsilon_Y(k)$ como las coordenadas $\mathbf{x}$ e $\mathbf{y}$ de $\mathbf{C}_r$, cuando es el contorno de un cuadrado centrado en (0,0), el índice de Poincaré podrá calcularse a través de la relación:

$$\mathbf{P}_{\Theta,\mathbf{C}}(i, j) = \sum_{k=0}^{N_{C_r}^2 - 1} \Delta_{ij}(k) \qquad (4.1)$$

Donde:

$$\Delta_{ij}(k) = \begin{cases} \delta_{ij}(k) & \text{si: } |\delta_{ij}(k)| < \pi/2 \\ \pi + \delta_{ij}(k) & \text{si : } \delta_{ij}(k) \le -\pi/2 \\ \pi - \delta_{ij}(k) & \text{para otro caso} \end{cases} \tag{4.2}$$

Tal que:

$$\delta_{ij}(k) = \Theta(i + \varepsilon_X(k+1), j + \varepsilon_Y(k+1)) - \Theta(i + \varepsilon_X(k), j + \varepsilon_Y(k)) \tag{4.3}$$

Se observa que $\mathbf{P}_{\Theta,C}(i,j)$ durante el recorrido asume únicamente los siguientes valores $0^o, \pm 180^o, \pm 360^o, \ldots..$ y dependiendo del valor asumido el punto (i,j) podrá ser catalogado como: un punto ordinario si es 0°, un punto singular *Core* si es 180° y *Delta* si es -180°, tal como muestra la figura 4.3.

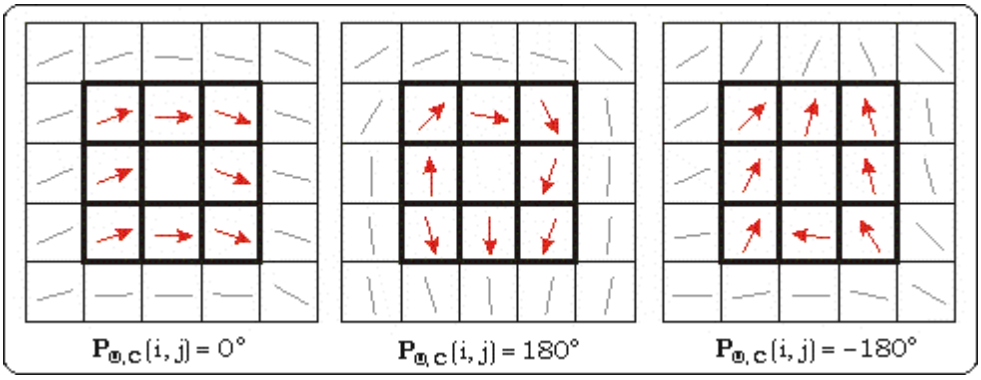


Figura 4.3 Cálculo del índice de Poincaré para r=1.

3. Luego de obtener $\mathbf{P}_{\Theta,C}$ para cada punto de la Imagen de Orientación se realiza un post procesamiento para eliminar aquellas falsas singularidades producidas por el ruido presente en la impresión dactilar, primero todas aquellas singularidades que pertenecen o están muy cercanas al segundo plano son descartadas. A las singularidades restantes se las agrupa a través del algoritmo de cuantización vectorial (*Vector Quantization* - VQ), [Zhang et al., 2001] los centroides generados por este algoritmo son asumidos como puntos singulares válidos (ver figura 4.4).

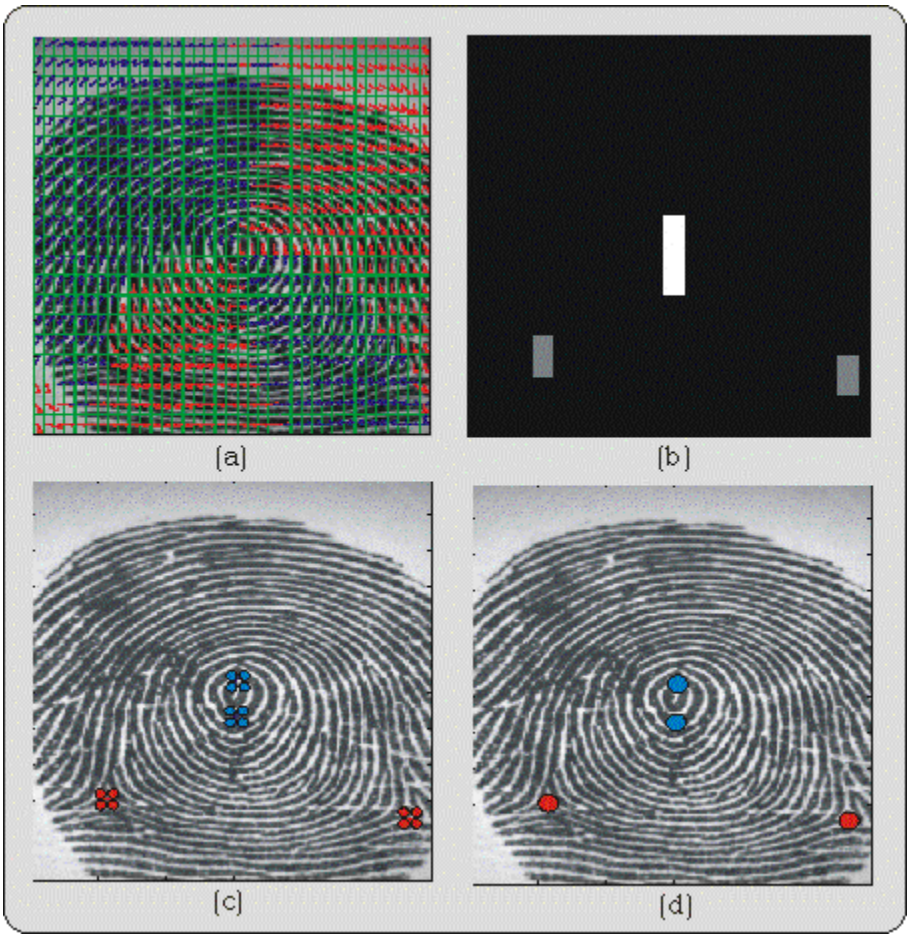


Figura 4.4 Detección de singularidades a través del índice de Poincaré (a) Imagen de Orientación suavizada; (b) índice de Poincaré; (c) singularidades determinadas (puntos azules son *Core* y rojos *Delta*); (d) singularidades válidas después de aplicar un algoritmo de agrupamiento. Imagen 18_6.tif de la base de datos 2 del FVC2002.

Generalmente el punto *Core* es empleado como punto de referencia. El número de puntos *Core* difiere de acuerdo a la clase a la que pertenece la impresión dactilar, los tipos TA, LL, y RL poseen un punto *Core*, mientras que los tipos TL y W poseen dos puntos *Core*, el tipo A no presenta un punto *Core*. Por consiguiente, se requerirán de dos técnicas de detección, una para los tipos TA, LL, TL y W, además otra para el tipo A. Para el primer caso usamos la técnica basada en el índice de Poincaré, estudiado en detalle en la sección anterior y para el segundo caso usamos la técnica basada en la transformación de covarianza de la Imagen de Orientación [Jain et al., 1999a]. El procedimiento se describe en dos pasos:

1. Determinamos los puntos *Core* de la impresión dactilar a través del índice de Poincaré. Si son detectados dos *Core*, el punto de referencia es asignado al *Core* superior. Si es detectado únicamente un *Core*, éste es asumido como el punto de referencia.
2. Si no se detectan puntos *Core*, se calcula la matriz de covarianza de cada vecindad local de Θ centrada en (i, j). Definimos una imagen característica conformada por los mayores eingenvalores de todas las matrices de covarianza calculadas. Un *Core* es detectado en el centroide de los mayores componentes conectados de la imagen característica umbralizada y el centro es asignado como el punto de referencia.

A continuación veamos en detalle los pasos requeridos para el cálculo de la transformación de covarianza de la Imagen de Orientación (ver figura 4.5).

1. Para cada punto (i, j) de Θ perteneciente a una región reconocible de la impresión dactilar, tomamos una sub matriz de dimensiones qxq $(q = 3)$ centrada en (i, j) denotada por Θ_{ij}. Todos los puntos de la sub matriz Θ_{ij} deben pertenecer a una región reconocible.

2. Calculamos la matriz de covarianza de Θ_{ij}. Utilizando para ello:

$$\Sigma_{ij} = \frac{1}{q-1}\sum_{k=1}^{q}(\Theta_{ij}^{k} - \mathbf{u}_{ij})\cdot(\Theta_{ij}^{k} - \mathbf{u}_{ij})^{T} \qquad (4.4)$$

Donde: Θ_{ij}^{k} denota la k-ésima columna de la sub matriz Θ_{ij} y $\mathbf{u}_{ij}$ denota el vector promedio de las columnas de la sub matriz Θ_{ij}.

3. Luego determinamos el máximo eingenvalor de Σ_{ij} [Mathews y Fink, 1999, pp. 629-640]. El valor de la transformación de covarianza en (i, j) será el máximo eingenvalor correspondiente a la matriz de covarianza Σ_{ij}, denotado por λ_{ij}.

4. Los pasos del 1 al 3 son repetidos para todos aquellos i y j pertenecientes a una región reconocible, obteniendo así la transformación de covarianza, representada por la matriz Ψ.

5. $\boldsymbol{\Psi}$ es binarizada con un valor umbral igual a la sexta parte del máximo valor de la matriz $\boldsymbol{\Psi}$ [Bobung, 2002], obteniéndose la matriz $\mathbf{B}_{\Psi}$.

6. Buscamos los componentes conectados en la matriz $\mathbf{B}_{\Psi}$ [Gonzáles y Woods, 1996, pp. 43 - 46] considerando que el punto de referencia de la impresión dactilar es el centro geométrico del componente conectado de mayor área.

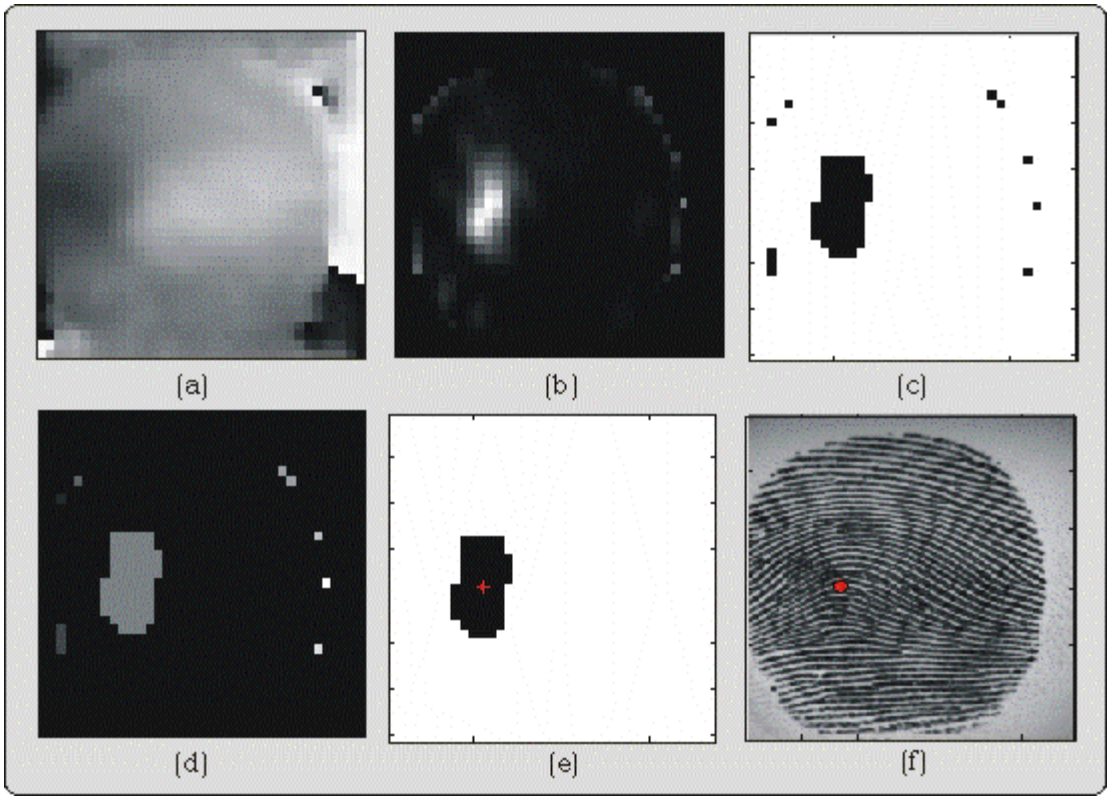


Figura 4.5 Detección del punto de referencia para imágenes de tipo A (a) Imagen de Orientación; (b) transformación de covarianza de la Imagen de Orientación; (c) imagen binaria de la transformación de covarianza; (d) etiquetado de la imagen binaria; (e) centro de masa del componente conectado de mayor área; (f) punto de referencia de la impresión dactilar. Imagen 10_5.tif de la base de datos 2 del FVC2002.

4.2.1.2. Técnica basada en los componentes sinusoidales de la Imagen de Orientación

Una nueva técnica para la detección de los puntos singulares usando las curvas de cambio de signo de la Imagen de Orientación fue propuesta por [Ramö et

al., 2001]. Este algoritmo analiza a Θ través de su componente senoidal y cosenoidal, definiendo como puntos singulares a todo punto ubicado en la intersección de las curvas de nivel de ambos componentes, por lo que el paso de mayor importancia radica en la determinación fehaciente de estos puntos sin dar lugar a una ambigüedad. Ramö propone usar la relación angular entre los Vectores Gradiente de cada componente perteneciente a las curvas de nivel cero (curvas de cambio de signo); tal que, si el ángulo que forman es 90º estamos frente a un punto *Core*, si forman 270º estamos frente a un punto *Delta*, el cumplimiento de esta condición se puede apreciar en la figura 4.6.

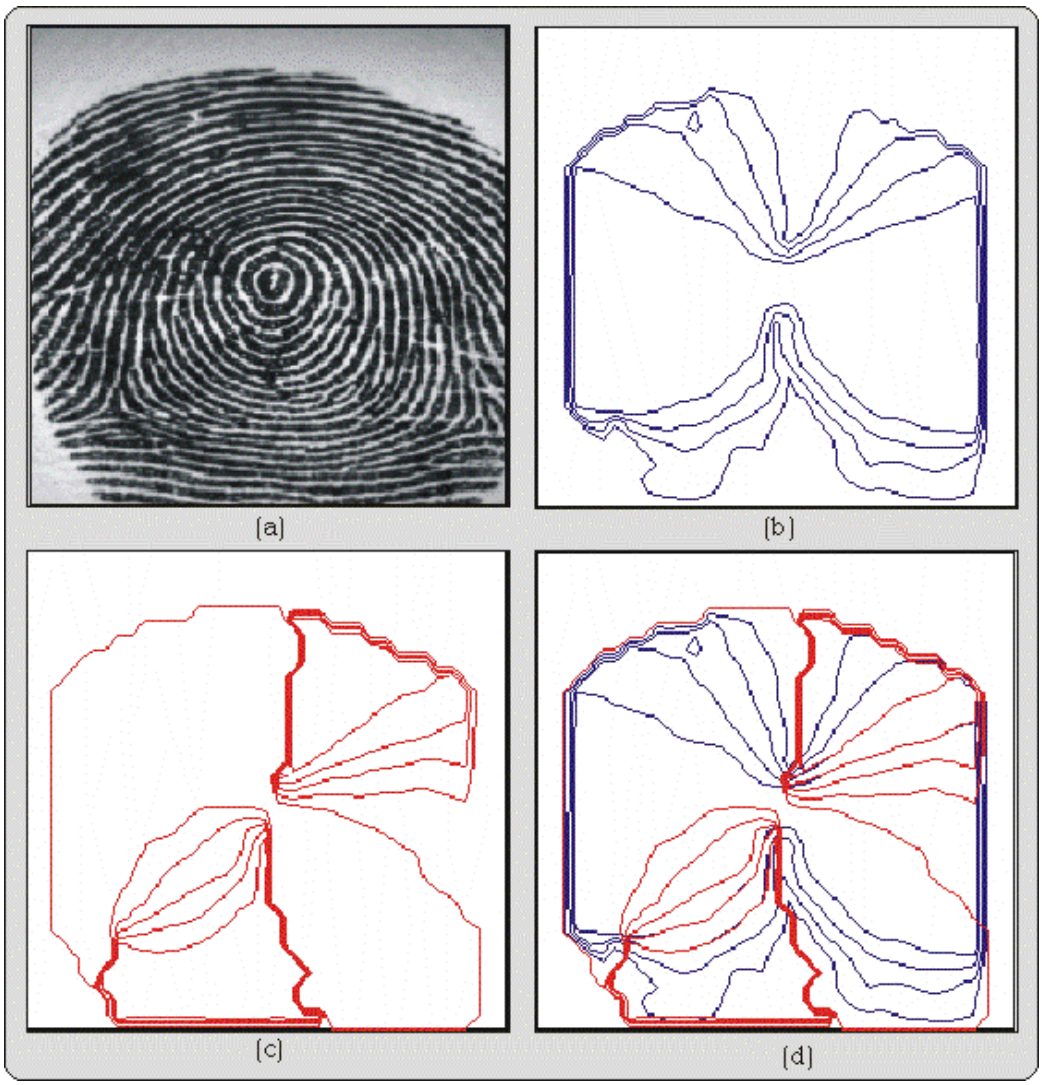


Figura 4.6 (a) Imagen original; (b) Curvas de nivel del componente senoidal; (c) curvas de nivel del componente cosenoidal; (d) superposición de curvas de nivel. Imagen 18_6.tif de la base de datos 2 del FVC2002.

Veamos a continuación los pasos requeridos para la detección de los puntos singulares utilizando esta técnica:

1. Determinamos el campo vectorial continuo de Θ a través de las relaciones:

$$\Theta_{\mathbf{X}}(i, j) = \cos(\Theta(i, j)) \tag{4.5}$$

$$\Theta_{\mathbf{Y}}(i, j) = \operatorname{sen}(\Theta(i, j)) \tag{4.6}$$

2. Determinamos la magnitud del Vector Gradiente de $\Theta_{\mathbf{X}}$, utilizando su aproximación discreta.

$$\begin{aligned} \left\|\nabla\Theta_{\mathbf{X}}(i, j)\right\|^2 &\approx \left|\frac{\partial\Theta_{\mathbf{X}}(i, j)}{\partial i}\right| + \left|\frac{\partial\Theta_{\mathbf{X}}(i, j)}{\partial j}\right| \\ &= \left|\Theta_{\mathbf{X}}(i+1, j) - \Theta_{\mathbf{X}}(i-1, j)\right| + \left|\Theta_{\mathbf{X}}(i, j+1) - \Theta_{\mathbf{X}}(i, j-1)\right| \end{aligned} \tag{4.7}$$

3. Para todo punto (i, j) que pertenezca al primer plano y $\left\|\nabla\Theta_{\mathbf{X}}(i, j)\right\|^2 \neq 0$, calculamos el seno del ángulo que forman los Vectores Gradiente de $\Theta_{\mathbf{X}}$ y $\Theta_{\mathbf{Y}}$ utilizando la multiplicación vectorial "$\circ$" definida como: $\mathbf{a} \circ \mathbf{b} = \mathbf{a}_{\mathbf{x}}.\mathbf{b}_{\mathbf{y}} - \mathbf{a}_{\mathbf{y}}.\mathbf{b}_{\mathbf{x}} = \|\mathbf{a}\|.\|\mathbf{b}\|\operatorname{sen}(\alpha)$; donde α es el ángulo formado por los vectores $\mathbf{a}$ y $\mathbf{b}$, entonces asumiendo que los Vectores Gradiente de ambos componentes son unitarios y utilizando la aproximación discreta del Vector Gradiente tendremos que:

$$\begin{aligned} \alpha_{\mathbf{XY}}(i, j) &= \nabla\Theta_{\mathbf{X}}(i, j) \circ \nabla\Theta_{\mathbf{Y}}(i, j) \\ &= \begin{vmatrix} \Theta_{\mathbf{X}}(i+1, j) - \Theta_{\mathbf{X}}(i-1, j) & \Theta_{\mathbf{Y}}(i+1, j) - \Theta_{\mathbf{Y}}(i-1, j) \\ \Theta_{\mathbf{X}}(i, j+1) - \Theta_{\mathbf{X}}(i, j-1) & \Theta_{\mathbf{Y}}(i, j+1) - \Theta_{\mathbf{Y}}(i, j-1) \end{vmatrix} \end{aligned} \tag{4.8}$$

En la figura 4.7 se muestra a $\alpha_{\mathbf{XY}}$ como una superficie que presenta picos máximos y mínimos, aquellos picos máximos corresponden a las posiciones vinculadas a los puntos *Core* $(\operatorname{sen}(90^\circ) = 1)$ y los picos mínimos corresponde a las posiciones vinculadas a los puntos *Delta* $\operatorname{sen}(270^\circ) = -1$.

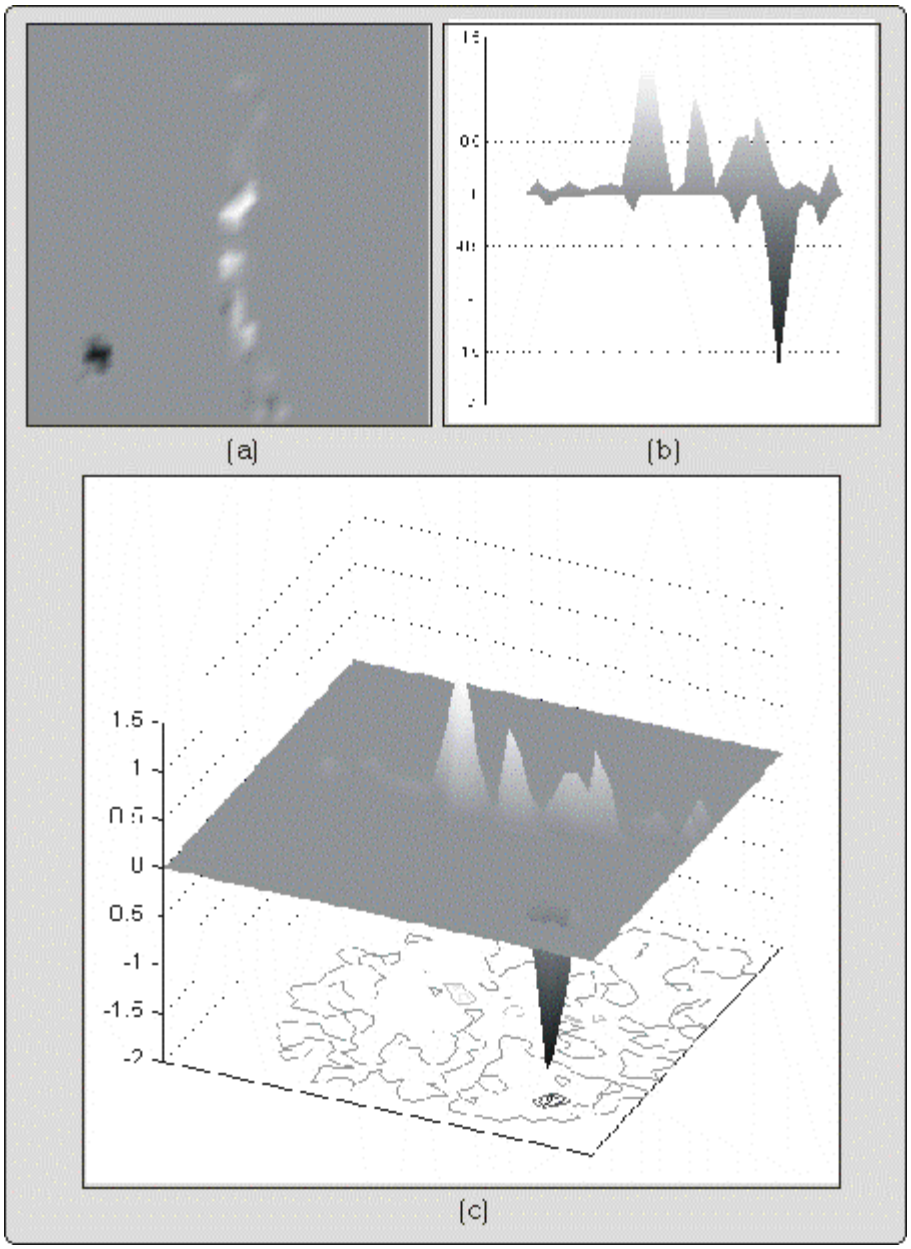


Figura 4.7 Vistas de α_{XY} (a) como imagen (negro es mínimo blanco es máximo); (b) vista frontal; (c) como superficie.

4. Tomando en cuenta las características que presenta la superficie α_{XY} podremos establecer la presencia de un *Core* o un *Delta* para todo punto (i,j) si se cumple la relación:

$$\square\alpha_{XY}(i,j)\square = \begin{cases} 1 & (i,j) \text{ es } \textit{Core} \\ -1 & (i,j) \text{ es } \textit{Delta} \end{cases} \qquad (4.9)$$

En la figura 4.8 se muestran las imágenes vinculadas a cada uno de los pasos para la detección de los puntos singulares a través de los componentes sinusoidales de Θ.

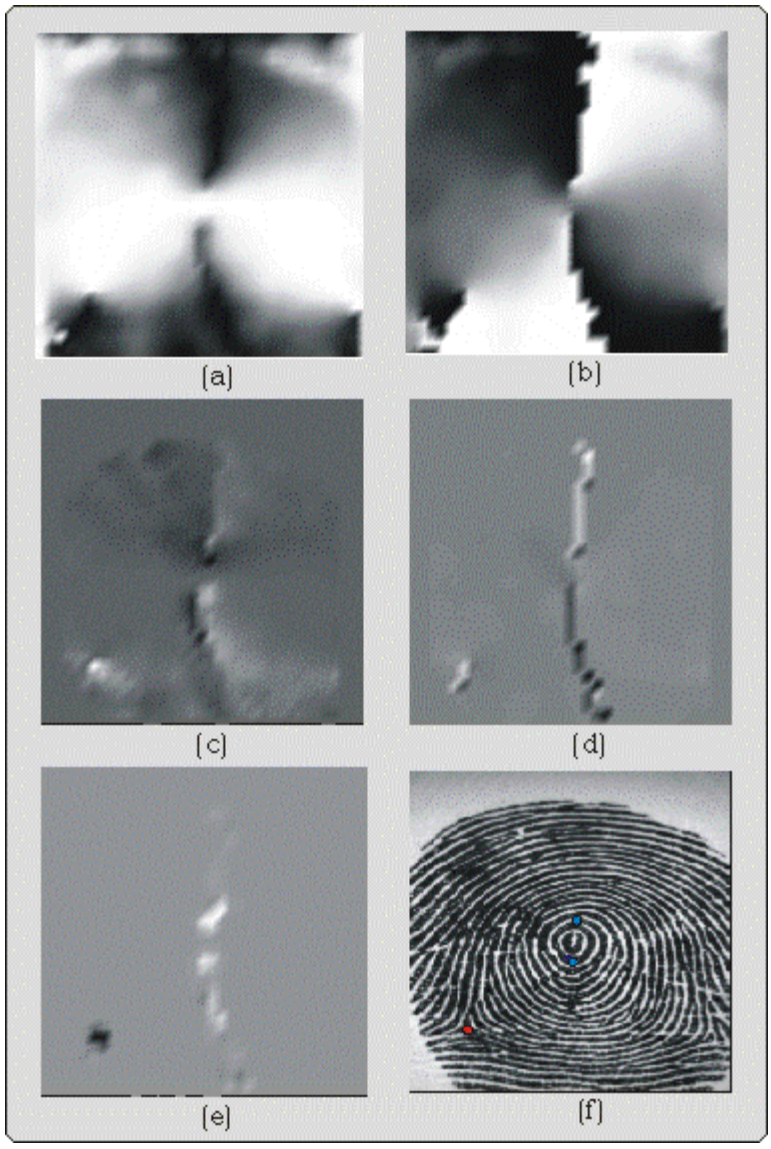


Figura 4.8 Obtención de los puntos singulares a través de los componentes sinusoidales de Θ. (a) Θ_Y; (b) Θ_X; (c) $\|\nabla\Theta_Y\|$; (d) $\|\nabla\Theta_X\|$; (e) α_{XY}; (f) singularidades determinadas (puntos azules *Core* y rojos *Delta*). Imagen 18_6.tif de la base de datos 2 del FVC2002.

Para la determinación del punto de referencia se toman las mismas consideraciones que para la técnica basada en el índice de Poincaré (el punto *Core* superior es asumido como punto de referencia), sin embargo esta técnica tiene

como ventaja adicional la capacidad de detectar el punto de referencia de una impresión dactilar tipo A en forma directa.

4.3. Extracción de las características locales

Las características locales de mayor importancia son las minucias pues son ampliamente usadas en la etapa de coincidencia, éstas forman una representación compacta que captura un componente significativo de información individual. Comparado a otras características, las minucias son relativamente más robustas ante las variadas fuentes de degradación de las impresiones dactilares. La mayoría de los tipos de minucias no son estables y no pueden ser identificados confiablemente por técnicas automáticas de procesamiento de imágenes. Por consiguiente, para la coincidencia de impresiones dactilares en forma automática sólo los dos más prominentes tipos de minucias son usados debido a su estabilidad y robustez: las minucias de finalización y bifurcación. Típicamente una impresión dactilar de alta calidad contiene cerca de setenta a cien minucias. La representación de una impresión dactilar según el estándar ANSI-NIST está basada en las minucias de finalización y bifurcación incluyendo su localización y dirección.

4.3.1. Extracción de las minucias

La fidedigna extracción de las minucias desde la imagen de entrada es un paso crítico para todo sistema de reconocimiento, puesto que el rendimiento del algoritmo de extracción de minucias depende principalmente de la calidad de la imagen de entrada. En una impresión dactilar ideal, los valles y crestas se alternan fluyendo en una dirección local constante y las minucias son anormalidades de las crestas. En tales situaciones las crestas pueden ser fácilmente detectadas y las minucias pueden ser localizadas con precisión desde la imagen binarizada de la impresión dactilar. Sin embargo, en la práctica debido a las variaciones en las condiciones de la impresión, configuración de las crestas, estado de las superficies de las yemas (marcas post natales, formaciones aberrantes en la epidermis de la huella, entre otras), dispositivos de adquisición, actitudes no cooperativas de los titulares de la huella dactilar, entre otros; un gran porcentaje de las impresiones

dactilares adquiridas son de pobre calidad, lo cual genera los siguientes problemas: (a) pueden ser creadas un significativo número de minucias espurias; (b) pueden ser ignoradas un gran porcentaje de minucias genuinas; (c) pueden presentarse errores en sus localizaciones (posición y dirección). Para que el algoritmo de extracción de minucias sea robusto con respecto a la calidad de la imagen de entrada "debe incluirse un algoritmo para el mejoramiento", teniendo éste como propósito mejorar la claridad de las estructuras en las crestas. Se ha desarrollado un algoritmo de extracción de minucias basado en los trabajos de [Watson et al., 1994], [Willis y Myers, 2001], [Tico et al., 2001], [Tico y Onnia, 2002], [Zang y Suen, 1984], [Zhao y Tang, 2002] y [Jain et al., 1997b]. En las siguientes secciones describimos en detalle el algoritmo de extracción de minucias. Asumimos que la resolución de la impresión dactilar de entrada es 500 dpi. El diagrama de flujo del algoritmo de extracción de minucias es mostrado en la figura 4.9. Los pasos principales son:

1. **Mejora de la imagen**: La imagen original es mejorada a través de la ecualización del histograma y la aplicación de la transformada discreta de Fourier bidimensional en forma local.
2. **Binarización y esqueletización de la imagen**: La imagen mejorada es binarizada utilizando la información brindada por la segunda derivada direccional de los píxeles de la imagen. Luego cada cresta binarizada es adelgazada hasta que tenga como máximo un píxel de ancho.
3. **Extracción de las minucias**: A partir de la imagen binaria esqueletizada se extrae la posición de las minucias utilizando el índice de cruces por ceros y se realiza una etapa de post procesamiento para eliminar las falsas minucias.

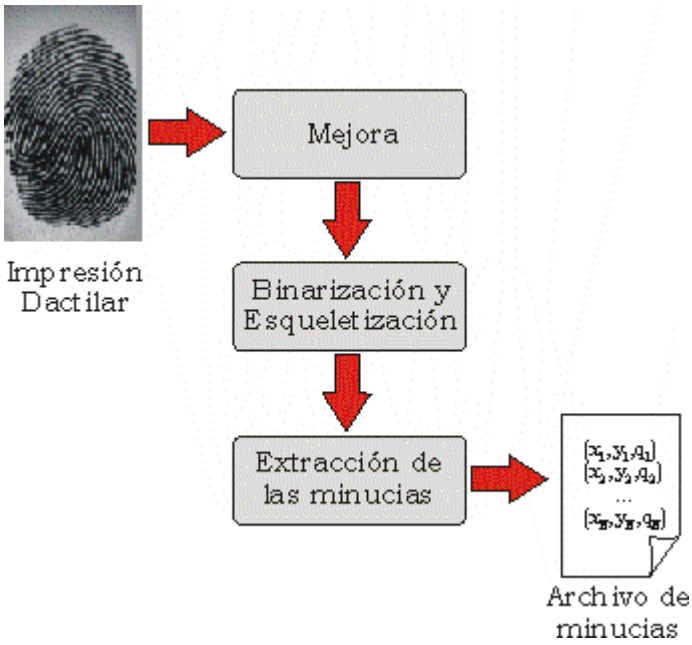


Figura 4.9 Diagrama de flujo para el algoritmo de extracción de minucias.

Veamos en detalle cada uno de estos pasos:

4.3.1.1. Mejora de la imagen

El principal objetivo es obtener una imagen resultante más adecuada que la original para la binarización, de manera que se eliminen algunos efectos no deseados y que las crestas presenten un mayor contraste que los valles. Dividimos el proceso de mejora en dos fases:

1. **Realzado del contraste**: Ecualizamos el histograma de los píxeles pertenecientes al primer plano y desplazamos el histograma resultante para resaltar las crestas sobre los valles.
2. **Mejora a través de la transformada de Fourier bidimensional**: La imagen resultante es sectorizada en bloques y para cada bloque se determina la transformada de Fourier bidimensional correspondiente. El mejoramiento se realiza utilizando el módulo de esta transformada. Es esencial considerar cierto traslapamiento entre bloques para evitar las discontinuidades entre bloques adyacentes.

4.3.1.1.1. Realzado del contraste

La idea subyacente en las técnicas de realzado de contraste es la variación del rango dinámico de los niveles de intensidad de la imagen que se está procesando. Una forma de realizar dicha variación es a través de la modificación del histograma de la imagen. El histograma de una imagen $h(r)$ es una función discreta que representa el número de píxeles de la imagen en función de sus niveles de intensidad (de 0 a 255), es decir:

$$h(r) = \frac{N_r}{N} \quad ; r = \{0,1,\cdots,255\} \tag{4.10}$$

Donde: N_r es el número de píxeles de la imagen que toman el r-ésimo nivel de intensidad; N es el número total de píxeles en la imagen. De forma general, se puede decir que $P(r)$ da una idea del valor de la probabilidad que aparezca el nivel de intensidad r. La representación gráfica de esta función para todos los valores de r proporciona una descripción global de la apariencia de una imagen, por ejemplo: un histograma con una distribución de los niveles de grises concentrada en una determinada zona presenta un contraste muy bajo, mientras que un histograma con un amplia distribución de los niveles de grises tiene un alto contraste. Un histograma con los niveles de grises concentrados en la parte baja del rango corresponde a una imagen oscura, un histograma con los valores concentrados en la parte alta del mismo corresponden a una imagen brillante. Una de las técnicas más utilizadas para la modificación del histograma es denominada igualación o ecualización uniforme del histograma ([Gonzáles y Woods, 1992, pp. 195-198], [Pajares y De la Cruz, 2002, pp. 105 - 112]). Se trata de encontrar una función que realce el contraste general de la imagen original expandiendo la distribución de los niveles de grises. Dicha distribución debe ser lo más suave posible en el sentido de que idealmente debería haber el mismo número de píxeles por niveles de intensidad. Es decir, el objetivo es distribuir los niveles de intensidad de una manera uniforme a lo largo de todo el rango de valores de niveles de intensidad. La ecualización del histograma se realiza trabajando sobre el histograma acumulado, definido como:

$$H(r) = \sum_{k=0}^{r} h(k) \quad ; r = \{0,1,\cdots,255\} \tag{4.11}$$

Si el histograma fuese totalmente plano, la función de densidad de probabilidad supuesta sería:

$$h_{pl}(r') = \frac{1}{255}; \quad r' = \{0,1,\cdots,255\} \tag{4.12}$$

Lo que genera el histograma acumulado:

$$H_{pl}(r') = (r'+1)\frac{N}{256} \tag{4.13}$$

Y como lo que deseamos es uniformizar el histograma, entonces tendremos: $H(r) = H_{pl}(r')$, lo que implica:

$$r' = \frac{256}{N} H(r) - 1 \tag{4.14}$$

Como los niveles de grises son valores enteros se realiza un redondeo a través de la relación:

$$r_{nuevo} = \left\lfloor \frac{256}{N} H(r_{antiguo}) - 1 \right\rfloor \tag{4.15}$$

Para nuestra aplicación tomamos como histograma de referencia el generado por los píxeles pertenecientes al primer plano, ya que únicamente se quiere uniformizar el contraste entre crestas y valles. Luego se desplaza el histograma resultante negativamente (hacia la parte oscura) resaltándose así las crestas sobre los valles. De lo expuesto, es obvio que se requiere como entrada la imagen original de la impresión dactilar $\mathbf{I}$ y la Imagen de Regiones reconocibles $\mathfrak{R}$. En la figura 4.10 se muestra la imagen realzada resultante considerando una ecualización uniforme y un desplazamiento negativo de 10 (niveles de intensidad).

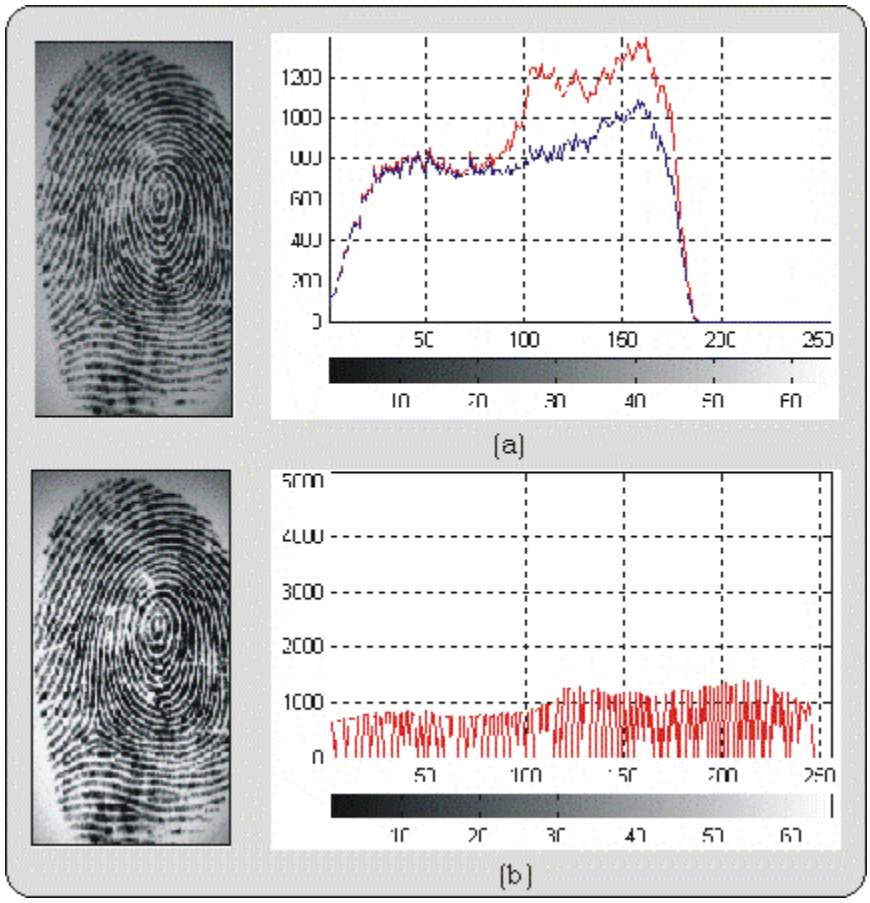


Figura 4.10 Realzado del contraste (a) imagen original y su respectivo histograma, la curva de color rojo corresponde al histograma de la imagen entera y la azul únicamente al histograma de la región de la imagen perteneciente al primer plano; (b) imagen resultante y su histograma ecualizado. Imagen 18_1.tif de la base de datos 2 del FVC2002.

4.3.1.1.2. Mejora a través de la transformada de Fourier bidimensional

Puesto que una impresión dactilar está conformada por estructuras de crestas y valles, la información direccional es de gran interés. En el dominio de la frecuencia la información direccional está contenida en el módulo de la transformada de Fourier bidimensional. Las técnicas de mejora que aprovechan esta característica fueron propuestas por [Watson et al., 1994] y [Willis y Myers, 2001]. En ambas la imagen es sectorizada en bloques de $N_B x N_B$ píxeles, con un nivel de traslapamiento de N_{TR} píxeles por lado (el traslapamiento entre bloques de la

imagen de entrada reduce las discontinuidades entre bloques adyacentes en la imagen mejorada); para cada bloque traslapado se determina su representación en el dominio de la frecuencia a través del cálculo de la transformada rápida de Fourier bidimensional (*Two-dimensional Fast Fourier Transform* - FFT2) [Gonzáles y Woods, 1996, pp. 129-138], luego se aplica una función no lineal a los datos resultantes que tiende a incrementar la "potencia" de la información útil (el patrón global y la orientación particular de las crestas y valles) frente al ruido. Finalmente para retornar al dominio espacial se calcula la transformada inversa de Fourier bidimensional (*Two-dimensional Inverse Fast Fourier Transform* - IFFT2) y procedemos a descartar los N_{TR} píxeles por lado (traslapamiento) obteniendo los datos mejorados del bloque en cuestión. El procedimiento de mejora puede ser resumido a través de la siguiente ecuación:

$$\mathbf{I}_{Mej}(u,v) = \left\| IFFT2\left\{ \left\| FFT2\{\mathbf{I}(u,v)\} \right\|^{k} .FFT2\{\mathbf{I}(u,v)\} \right\} \right\|;\ (u,v) \in \mathbf{S}_{ij}^{N_B,N_{TR}} \qquad (4.16)$$

Donde: $\mathbf{S}_{ij}^{N_B,N_{TR}}$ es el bloque (i,j)-ésimo vinculado a $\mathbf{I}$ de tamaño $N_B x N_B$ y con un traslapamiento de N_{TR} píxeles; $\|\bullet\|$ representa el operador módulo de un número complejo; k es el exponente al que es elevado la densidad espectral de energía [Oppenheim et al., 2000, pp. 60-64] del bloque $\mathbf{S}_{ij}^{N_B,N_{TR}}$, en [Watson et al., 1994] se estableció $k = 0.6$ y en [Willis y Myers, 2001] se propuso $k = 1.4$. El tiempo de procesamiento está directamente vinculado con el tamaño del bloque y el nivel de traslapamiento, por lo tanto no es aconsejable reducir el traslapamiento puesto que genera distorsiones entre bordes de cada bloque, ni tampoco aumentarlo porque incrementa el tiempo de procesamiento. En la figura 4.11 se muestran imágenes mejoradas bajo esta técnica considerando $N_B = 40$, $N_{TR} = 20$ y $k = 0.7$.

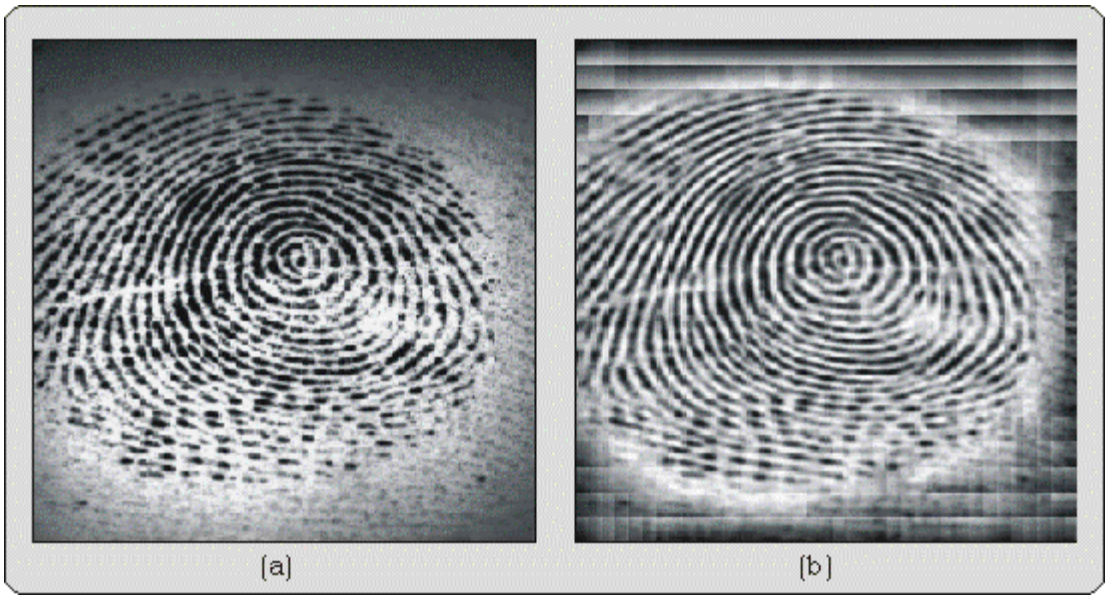


Figura 4.11 Mejora de la imagen a través de la transformada de Fourier bidimensional (a) imagen original; (b) imagen mejorada. Imagen 101_8.tif de la base de datos 2 del FVC2002.

4.3.1.2. Binarización y esqueletización de la imagen

Las técnicas de binarización y esqueletización desarrolladas son:

1. **Binarización direccional**: Se binariza la imagen utilizando como información de entrada la Imagen de Orientación y la Imagen de Regiones, además se aprovecha la información brindada por la segunda derivada direccional de los píxeles de la imagen.
2. **Esqueletización**: Se emplea un algoritmo de adelgazamiento para obtener la huella esquelética desde la imagen binaria.

Veamos a continuación cada una de estas técnicas.

4.3.1.2.1. Binarización direccional

En ciertas aplicaciones las imágenes en escala de grises requieren ser convertidas a imágenes binarias (negro/blanco). Esta operación es referida en la literatura como binarización o umbralización. La técnica estándar de binarización realiza la comparación numérica de los píxeles con un valor umbral. Si el píxel de interés tiene un valor menor al umbral, entonces el píxel correspondiente en la

imagen binaria toma el valor cero, en caso contrario toma el valor uno. La determinación de un valor umbral óptimo es un problema muy importante en el procesamiento de imágenes y se han propuesto diferentes soluciones [De la Escalera, 2001, pp. 199 - 214]. Para el caso de impresiones dactilares, la técnica más empleada se denomina umbralización adaptativa ([Farina et al., 1998] y [Verma et al., 1987]), en donde la imagen es sectorizada en bloques y para cada bloque se asigna como valor umbral el promedio de las intensidades de los píxeles. La limitante de esta técnica es su alta dependencia con la calidad de la imagen a binarizar. Para superar este inconveniente, usamos la binarización direccional presentada en [Tico et al., 2001] y [Tico y Onnia, 2002]. Esta técnica se basa en el hecho que los píxeles conformantes de una cresta son aquellos cuya intensidad presenta una segunda derivada direccional positiva. El cálculo de las derivadas direccionales en el píxel (u,v) perteneciente al bloque $\mathbf{S}_{ij}^{k}$, respecto al vector $\mathbf{v} = \left[\cos(\Theta(i,j)) \quad \text{sen}(\Theta(i,j))\right]^{T}$ viene dado por la expresión:

$$\begin{bmatrix} \dfrac{\partial \mathbf{I}(u,v)}{\partial p} \\ \dfrac{\partial \mathbf{I}(u,v)}{\partial q} \end{bmatrix} = \begin{bmatrix} \cos(\Theta(i,j)) & -\text{s en}(\Theta(i,j)) \\ \text{sen}(\Theta(i,j)) & \cos(\Theta(i,j)) \end{bmatrix} . \begin{bmatrix} \dfrac{\partial \mathbf{I}(u,v)}{\partial x} \\ \dfrac{\partial \mathbf{I}(u,v)}{\partial y} \end{bmatrix} \qquad (4.17)$$

La derivada parcial direccional de orden superior, respecto al vector $\mathbf{v}$ será:

$$\frac{\partial^2 \mathbf{I}(u,v)}{\partial p^2} = \frac{\partial^2 \mathbf{I}(u,v)}{\partial x^2} . \text{sen}^2(\Theta(i,j)) + \frac{\partial^2 \mathbf{I}(u,v)}{\partial y^2} . \cos^2(\Theta(i,j)) - \frac{\partial^2 \mathbf{I}(u,v)}{\partial x \partial y} . \text{sen}(2.\Theta(i,j)) \qquad (4.18)$$

Si denotamos las derivadas parciales respecto a los ejes coordenados a través de la siguiente notación: $\mathbf{D}^{(2,0)}(u,v) = \partial^2 \mathbf{I}(u,v) / \partial x^2$, $\mathbf{D}^{(0,2)}(u,v) = \partial^2 \mathbf{I}(u,v) / \partial y^2$ y $\mathbf{D}^{(1,1)}(u,v) = \partial^2 \mathbf{I}(u,v) / \partial x \partial y$, éstas podrán ser obtenidas por la función de interpolación:

$$\mathbf{D}^{(m,n)}(u,v) = \sum_{c=-L}^{L} \text{POL}_{(m,L)}(c) \left\{ \sum_{r=-L}^{L} \mathbf{I}(u-c, v-r) . \text{POL}_{(n,L)}(r) \right\} \qquad (4.19)$$

Donde:

$$POL_{(0,L)}(z) = \frac{3.(3L^2 + 3L - 1 - 5z^2)}{(2L+1).(2L-1).(2L+3)} \tag{4.20}$$

$$POL_{(1,L)}(z) = -\frac{5z.(15(L^3 + 2L^2 - 1).L - 7(3L^2 + 3L - 1)z^2 + 5)}{L.(L+1).(2L+1).(L-1).(2L-1).(L+2).(2L+3)} \tag{4.21}$$

$$POL_{(2,L)}(z) = -\frac{30.(z^2 - L^2 - L)}{L.(L+1).(2L+1).(2L-1).(2L+3)} \tag{4.22}$$

Son los polinomios de interpolación de Chebyshev [Mathews y Fink, 1999, pp. 250-262]; el valor de L se obtiene mediante la relación $L = 0.5T$, donde T es la distancia promedio entre crestas, normalmente es un valor casi constante para todas las crestas (para imágenes tomadas a 500 dpi T es aproximadamente ocho píxeles). Siendo $\partial^2 \mathbf{I}(u,v)/\partial p^2 > 0$ la condición que cumple todo píxel (u,v) perteneciente a una cresta [Serrat et al., 2000], la imagen binaria resultante viene dada por la expresión:

$$\mathbf{I}_{Bin}(u,v) = \begin{cases} 1 & \dfrac{\partial^2 \mathbf{I}(u,v)}{\partial p^2} > 0 \text{ ; es cresta} \\ 0 & \dfrac{\partial^2 \mathbf{I}(u,v)}{\partial p^2} < 0 \text{ ; es valle} \end{cases} \tag{4.23}$$

En la figura 4.12 se aprecia a la imagen inicial $\mathbf{I}$, la imagen mejorada y las imágenes binarizadas bajo ambas técnicas (adaptativa y direccional).

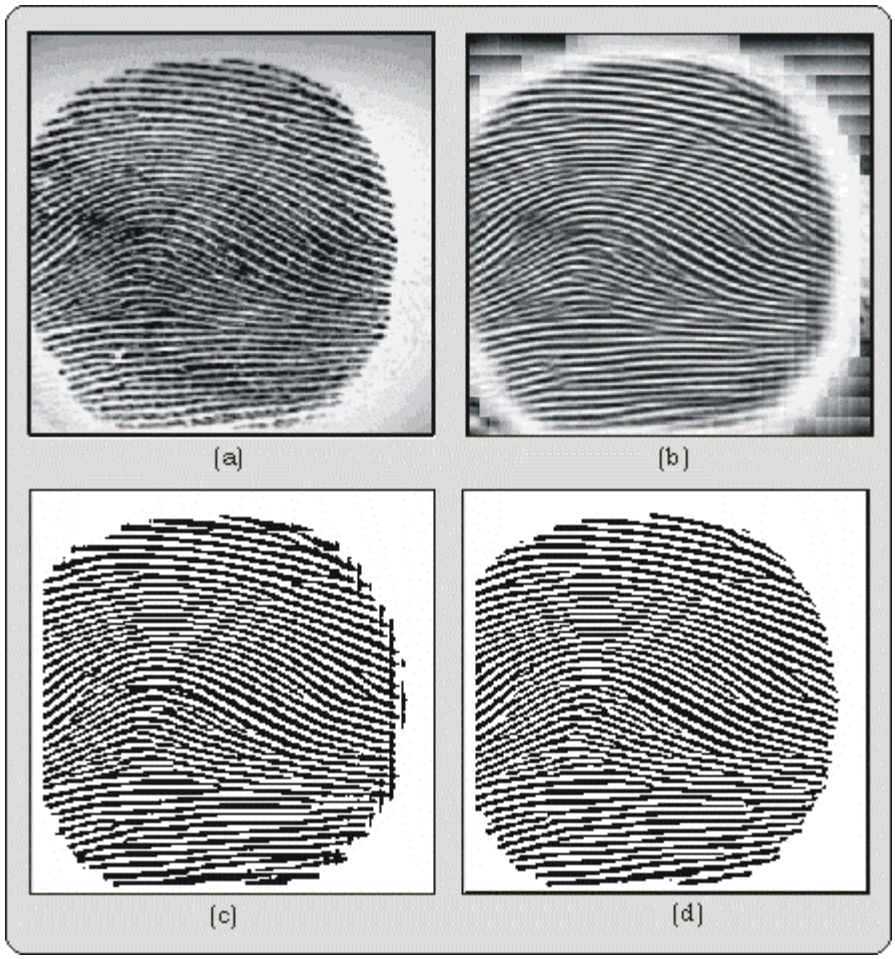


Figura 4.12 (a) Imagen original; (b) imagen mejorada; (c) imagen resultante de la binarización adaptiva; (c) imagen resultante de la binarización direccional. Imagen 10_5.tif de la base de datos 2 del FVC2002.

4.3.1.2.1. Esqueletización

La esqueletización busca representar una región u objeto por su grafo o esqueleto. El esqueleto de una región es un conjunto de puntos conectados, ubicados a la mitad de los bordes, teniendo éstos las siguientes propiedades: (a) presentan la misma conectividad que la región original; (b) el ancho del esqueleto es un píxel; (c) preserva los puntos aislados y de finalización de la región original. En la figura 4.13 mostramos el esqueleto de una sección de la imagen binaria de una impresión dactilar, puede apreciarse que se considera como regiones u objetos a las crestas binarizadas y el resto como fondo.

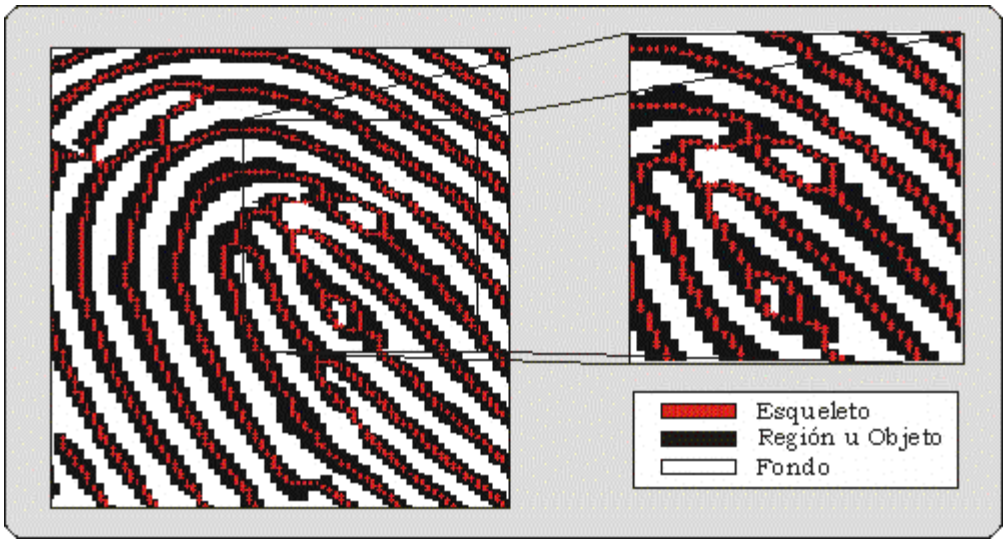


Figura 4.13 Esqueleto de una sección (en rojo) de una imagen binaria de una impresión dactilar.

Una manera de obtener el esqueleto está basada en la denominada transformación del eje medial (*Medial Axis Transformation* - MAT). Para cada elemento de la región se busca el punto de borde más cercano, si más de un punto del borde está a esa distancia el elemento pertenece al eje medial que se equipara al esqueleto. Como puede deducirse el costo computacional es prohibitivo. Por ello se han desarrollado otros algoritmos que consigan prácticamente el mismo fin, pero de forma más rápida, tal que para no llegar a destruir el esqueleto cumplen las siguientes condiciones: (a) no eliminar los puntos extremos; (b) no romper la conectividad; (c) no causar excesiva erosión en la región. Uno de ellos (el implementado en este trabajo) presentado en [Gonzáles y Woods, 1996, pp. 532-535], consiste en la aplicación sucesiva de dos pasos básicos a cada punto del contorno de una región dada, siendo un punto de contorno cualquier píxel perteneciente a una región que tenga al menos un 8-vecino que pertenece al fondo (la imagen debe estar binarizada con valor uno para los píxeles pertenecientes a una región y cero para los píxeles pertenecientes al fondo). Para ello tomamos un píxel P y una máscara con sus 8-vecinos: $P_1, P_2, \cdots, P_8$ ordenados desde la posición central derecha recorrido en sentido antihorario, como se muestra en la figura 4.14:

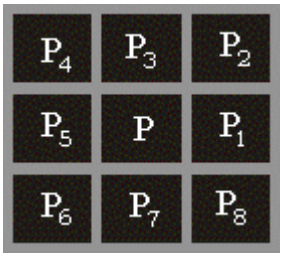


Figura 4.14 Vecindad alrededor del píxel P.

Paso 1: Todo píxel P de contorno $P_1, P_2, \cdots, P_8$ se etiqueta como fondo (elimina) si satisface las siguientes condiciones:

(a) $2 \le N(P) \le 6$

(b) $S(P) = 1$

(c) $P_1.P_3.P_7 = 0$

(d) $P_1.P_5.P_7 = 0$

Paso 2: Una vez eliminado todo píxel P que cumple con las condiciones del paso 1, se etiquetan como fondo aquellos píxeles P de contorno $P_1, P_2, \cdots, P_8$ que cumplen las siguientes cuatro condiciones:

(a) $2 \le N(P) \le 6$

(b) $S(P) = 1$

(c) $P_1.P_3.P_5 = 0$

(d) $P_3.P_5.P_7 = 0$

Donde:

$N(P)$: es el número de vecinos diferentes de cero en la vecindad centrada en P.

$S(P)$: es el número de transiciones entre ceros y unos, en la secuencia $P_1, P_2, \cdots, P_8$.

Note que los dos pasos anteriores coinciden en las primeras condiciones; las dos últimas indican un giro en la búsqueda de los píxeles a eliminar. Para ambos casos, la condición (a) no la cumplen (se preservan) los puntos extremos y la condición (b) asegura que el esqueleto tenga un píxel de ancho; las condiciones (c) y (d) la cumplen todo píxel que no pertenezca al esqueleto. Una manera eficiente de realizar este algoritmo es a través de una tabla de búsqueda [Zang y Suen, 1984]. Si se convoluciona la imagen con la máscara de codificación mostrada en la

figura 4.15, los 8-vecinos quedarán codificados en un número entre 0 y 255. Puede calcularse de antemano los valores numéricos que darían las condiciones expresadas anteriormente. Por ello se deben eliminar los puntos que cumplen:

$$\begin{aligned} Ps_1 = \{&3,6,7,12,14,15,24,28,30,48,56,60,62,96,112,\\ &120,129,131,135,143,192,193,195,199,207,\\ &224,225,227,231,240,241,243,248,249\} \end{aligned} \tag{4.24}$$

$$\begin{aligned} Ps_2 = \{&3,6,7,12,14,15,24,28,30,31,48,56,60,62,63,96,\\ &112,120,124,126,129,131,135,143,159,192,193,\\ &195,224,225,227,240,248,252\} \end{aligned} \tag{4.25}$$

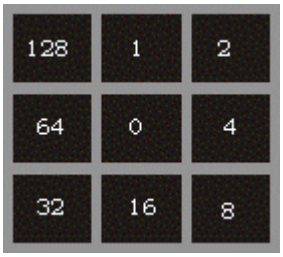


Figura 4.15 Máscara de codificación propuesta por [Zang y Suen, 1984].

Representando Ps_1 los posibles casos que cumplen el primer conjunto de condiciones y Ps_2 el segundo. En la figura 4.16 la imagen esqueletizada obtenida $\mathbf{I}_{esq}$ está superpuesta a la imagen original.

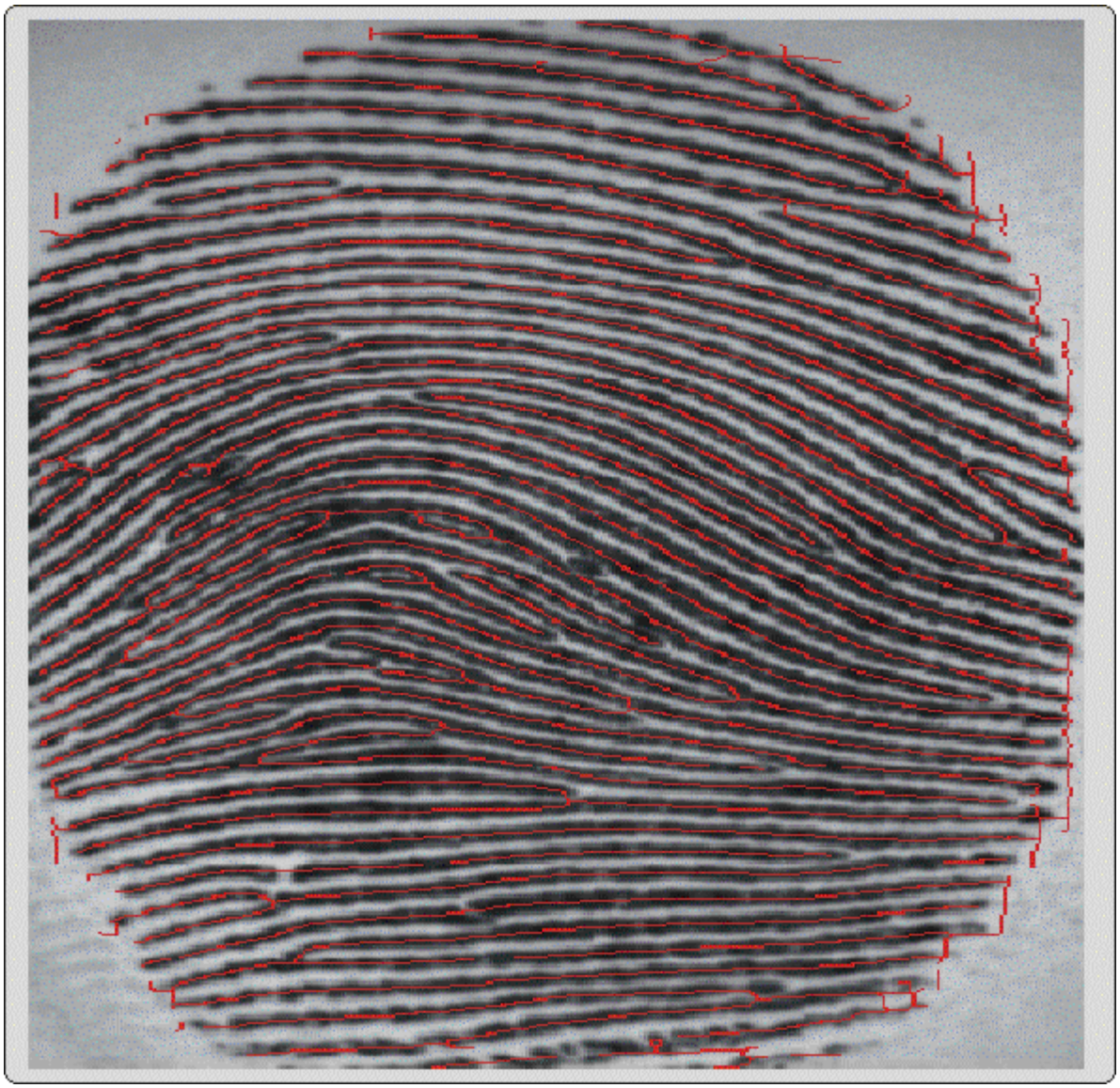


Figura 4.16 Imagen esqueletizada superpuesta a la imagen original. Imagen 10_5.tif de la base de datos 2 del FVC2002.

4.3.1.3. Extracción de las minucias y post procesamiento

La extracción de minucias es una tarea trivial cuando se cuenta con la imagen esqueletizada ideal. Este procedimiento generalmente consiste de los siguientes pasos:

1. **Pre procesamiento**: Permite eliminar todos los píxeles que puedan generar problemas al momento de detectar minucias sobre $\mathbf{I}_{esq}$.

2. **Detección de las minucias:** Determinar el tipo, la posición y la dirección de una minucia utilizando el número de cruces de Rutovitz y una técnica propuesta para el cálculo de su dirección.

3. **Post procesamiento**: Realizar un post procesamiento entre las minucias detectadas de acuerdo a algunas reglas heurísticas.

4.3.1.3.1. Pre procesamiento

Idealmente, el ancho de cada cresta adelgazada debe ser estrictamente un píxel. Sin embargo, esto no siempre es verdad; aquellos píxeles de $\mathbf{I}_{esq}$ que pertenezcan a una cresta con un ancho mayor a uno se denominan píxeles *bug* y se definen como aquellos píxeles con más de dos vecinos 4-conectados; su existencia genera el intercambio de los tipos de minucia y una mala detección de las bifurcaciones. Por consiguiente, antes de la detección de las minucias, es necesario implementar un algoritmo para la eliminación de los píxeles *bug*. Tal algoritmo es presentado en [Zhao y Tang, 2002] y consta de los siguientes pasos:

1. Hallamos todos los píxeles *bug* en $\mathbf{I}_{esq}$.

2. Retiquetamos como fondo el primer píxel *bug* hallado.

3. Determinamos si el siguiente píxel *bug* todavía presenta más de dos vecinos 4-conectados, si es así será vuelto a etiquetar como fondo, en caso contrario se preservará como un píxel objeto.

4. Volvemos al paso 3 hasta haber explorado todos los píxeles *bug* hallados inicialmente.

Algunos ejemplos son mostrados en la figura 4.17.

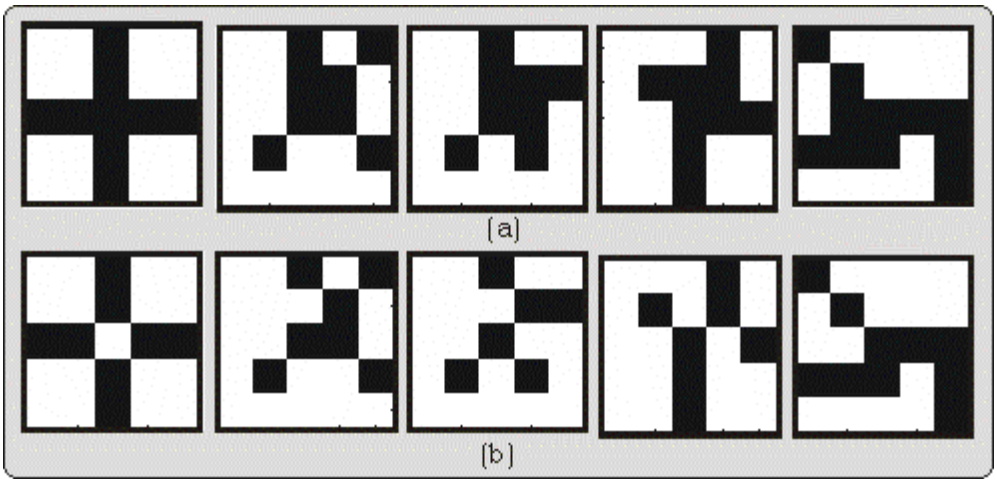


Figura 4.17 Ejemplos de píxeles *bug* y sus validaciones (a) sin procesar; (b) procesadas.

A pesar de haber eliminado los píxeles *bug,* $\mathbf{I}_{esq}$ aún presenta puntos dudosos que generan inconvenientes al momento de determinar otras características de las minucias tal como su dirección, para esta labor planteamos el filtrado de la imagen esquelética con un número determinado de máscaras de 3x3 píxeles que representan la mayoría de casos que generan dificultades al momento de determinar la dirección; éstas son mostradas en la figura 4.18.

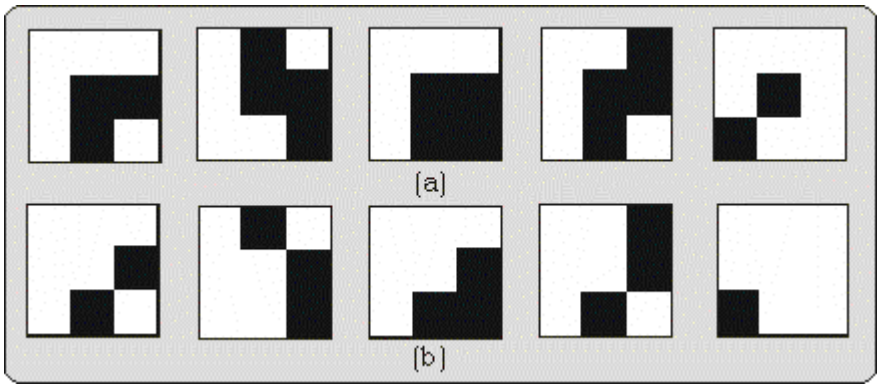


Figura 4.18 Puntos que generan problemas al momento de calcular la dirección (a) antes de eliminarlos; (b) después de eliminarlos .

Para lograr que esta tarea sea llevada a cabo rápidamente las máscaras y sus variaciones (tres rotaciones de 90º para cada máscara) quedarán codificadas en un número entre 0 y 255 al multiplicarlas con la máscara de codificación (ver figura 4.15), generándose las siguientes tablas:

$$Ps_1 = \{5,7,13,20,22,28,30,65,67,80,112,120,133,135,193,225\} \quad (4.26)$$

$$Ps_2 = \{2,8,5,20,32,52,65,80,88,97,128,208\} \quad (4.27)$$

Luego se procede a filtrar la imagen esquelética con la máscara de codificación, eliminando los puntos que cumplen con Ps_1. La imagen resultante es filtrada nuevamente con la máscara de codificación, eliminando esta vez los puntos que cumplen con Ps_2.

4.3.1.3.2. Detección de las minucias

La detección de las minucias tiene por finalidad determinar el tipo, la posición y la dirección de cada minucia, obteniéndose los dos conjuntos de tripletas:

$$\begin{aligned} MF &= \left\{(u_{F,1}, v_{F,1}, \theta_{F,1}); (u_{F,2}, v_{F,2}, \theta_{F,2}); \ldots; (u_{F,N_{MF}}, v_{F,N_{MF}}, \theta_{F,MF})\right\} \\ MB &= \left\{(u_{B,1}, v_{B,1}, \theta_{B,1}); (u_{B,2}, v_{B,2}, \theta_{B,2}); \ldots; (u_{B,N_{MB}}, v_{B,N_{MB}}, \theta_{B,MB})\right\} \end{aligned} \quad (4.28)$$

Éstas representan las posiciones y orientaciones de las minucias de finalización y bifurcación. La determinación del tipo y posición es realizada a través del cálculo del número de cruces y la determinación de la dirección es calculada con una técnica propuesta, veamos a continuación cada uno de estos pasos.

4.3.1.3.2.1. Determinación del tipo y posición

El concepto de número de cruces CN es extensamente usado para la determinación de la posición y el tipo de una minucia [Jain et al., 1997b]. La definición de Rutovitz del número de cruces para el píxel (u,v) es:

$$CN(u,v) = \frac{1}{2}\sum_{k=1}^{8} |P_i - P_{i+1}| \quad ; \{P_1, P_2, \cdots, P_8\} \in V(\mathbf{I}_{esq}(u,v)) \quad (4.29)$$

Asumimos que: $P_1 = P_9$. Dependiendo del valor que presente el indicador podremos catalogar a cada píxel P como un tipo específico de singularidad, tal como es mostrado en la tabla 4.1.

CN	PROPIEDAD
0	Punto Aislado
1	Punto de Finalización
2	Punto Conectado
3	Punto de Bifurcación
4	Punto de Cruce

Tabla 4.1 Singularidades especificadas por el CN.

El cálculo del CN es dependiente de la calidad de afinamiento obtenida para $\mathbf{I}_{esq}$, por lo que es necesario eliminar los píxeles *bug* antes de su determinación. Los pasos que involucran el cálculo del CN son:

1. Para todo punto perteneciente a una cresta se calcula el CN a través de la relación (4.29), si estamos sobre un punto perteneciente a un borde, aquellos valores de la vecindad extraída de 3x3 píxeles que no pertenecen a la imagen son fijados a cero.

2. Tomando los valores calculados de CN para cada punto perteneciente a una cresta determinamos el tipo y la posición de las minucias, obteniéndose dos conjuntos de puntos, el primero vinculado a las minucias de finalización y el segundo a las de bifurcación, tal como se muestra a continuación:

$$PF = \left\{(u_{F,1}, v_{F,1}); (u_{F,2}, v_{F,2}); \ldots; (u_{F,i}, v_{F,i}); \ldots; (u_{F,N_{MF}}, v_{F,N_{MF}})\right\}; CN(u_{F,i}, v_{F,i}) = 1 \qquad (4.30)$$

$$PB = \left\{(u_{B,1}, v_{B,1}); (u_{B,2}, v_{B,2}); \ldots; (u_{B,i}, v_{B,i}); \ldots; (u_{B,N_{MB}}, v_{B,N_{MB}})\right\}; CN(u_{B,i}, v_{B,i}) = 3 \qquad (4.31)$$

Lamentablemente debido a las perturbaciones que presenta una imagen, dentro de ambos conjuntos habrá minucias falsas y minucias válidas cuyo tipo ha sido asignado indebidamente, estos problemas son afrontados en la etapa de post procesamiento.

4.3.1.3.2.2. Determinación de la dirección

Aquí proponemos una técnica novedosa para la determinación de la dirección de una minucia, basada en un algoritmo de seguimiento de crestas "robusto", que tiene por finalidad la obtención de un conjunto de pares de coordenadas que definen una ruta recorrida sobre una cresta adelgazada de $\mathbf{I}_{esq}$. Diseñamos un algoritmo de seguimiento de crestas basado en una tabla de búsqueda que contiene todas las posibles opciones a tomar en una vecindad de 3x3 píxeles centrada en el "punto actual" donde se encuentre la ruta. La tabla tiene como entradas el "punto anterior" al actual en la vecindad de 3x3 píxeles, un código vinculado a la vecindad de 3x3 píxeles extraída y como salida devuelve el posible "punto siguiente". En la figura 4.19 se muestra todos los posibles tipos de vecindades de 3x3 píxeles que pueden presentarse, las letras A, C, S y X hacen referencia al "punto anterior", "punto actual" ,"punto siguiente" y un punto cualquiera que puede estar etiquetado como fondo o como objeto. Cada una de estas vecindades y sus variaciones (tres rotaciones de 90º para cada vecindad) quedarán codificadas entre 0 y 255 al multiplicarlas con la máscara de codificación, generándose una tabla de búsqueda de 256x8 opciones. La tabla es mostrada en el Apéndice A.

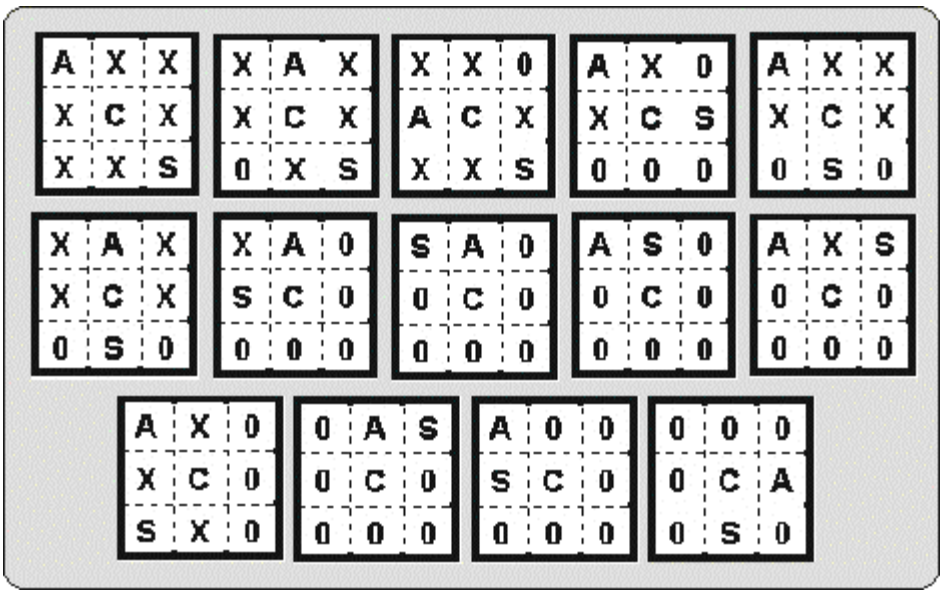


Figura 4.19 Máscaras que mapean todas las posibles opciones a tomar en una vecindad extraída de 3x3 píxeles centrada en el "punto actual" de la ruta.

Los pasos para el algoritmo de seguimiento de crestas son:

1. Establecemos el punto inicial de la ruta y lo definimos como "punto actual", extraemos una vecindad de 3x3 píxeles centrada en dicho punto, en ella buscamos el primer píxel etiquetado como objeto, comprobamos si es una minucia, en caso lo sea, el algoritmo de seguimiento finalizará teniendo la ruta una longitud de sólo un píxel; si no es una minucia, entonces será definido como el "punto siguiente".
2. Redefinimos el "punto actual" y el "punto siguiente" del paso anterior como "punto anterior" y "punto actual" del paso 2, extraemos una vecindad de 3x3 píxeles, centrada en el nuevo "punto actual", codificamos dicha vecindad al multiplicarla con la máscara de codificación obteniéndose un código entre 0 y 255; con la posición relativa del nuevo "punto anterior" respecto a la vecindad extraída y con el código obtenido se determina en la tabla de búsqueda cual será el posible "punto siguiente", comprobamos si el posible "punto siguiente" es una minucia, en caso lo sea, el algoritmo de seguimiento finalizará, si no es así, entonces será definido como el nuevo "punto siguiente".
3. Repetimos el paso 2 tantas veces como lo requiera el recorrido.

Ahora veamos los pasos a seguir para la determinar la dirección de las minucias de finalización:

1. Tomando como punto inicial a cada minucia de finalización se recorre la cresta a la que pertenece hasta llegar a un máximo de N_{RTS} posiciones recorridas o haber llegado a la posición de otra minucia, obteniéndose el conjunto de pares ordenados que definen la ruta:

$$RF_i = \left\{(u_{1,i}, v_{1,i}); (u_{2,i}, v_{2,i}); \ldots; (u_{L^i_{RF},i}, v_{L^i_{RF},i})\right\} \; ; 1 \le L^i_{RF} \le N_{RTS} \qquad (4.32)$$

Tal que: $(u_{F,i}, v_{F,i}) = (u_{1,i}, v_{1,i})$ y $i = 1, 2, \ldots, N_{MF}$

2. El cálculo de la dirección se obtiene a través de la relación:

$$\theta_{F,i} = \text{arctg2}\left(v_{1,i} - \frac{1}{L_{RF}^{i}} \sum_{k=1}^{L_{RF}^{i}} v_{k,i}, u_{1,i} - \frac{1}{L_{RF}^{i}} \sum_{k=1}^{L_{RF}^{i}} u_{k,i} \right) \quad (4.33)$$

Los pasos a seguir para el cálculo de la dirección para las minucias de bifurcación son:

1. Tomando como punto inicial a cada minucia de bifurcación recorremos cada una de las crestas en las que se bifurcan hasta llegar a un máximo de N_{RTS} posiciones recorridas o haber llegado a la posición de otra minucia, obteniéndose tres conjuntos de pares ordenados que definen cada una de las tres rutas:

$$RB_{i1} = \left\{ (u_{1,i1}, v_{1,i1}); (u_{2,i1}, v_{2,i1}); \ldots; (u_{L_{RB}^{i1},i1}, v_{L_{RB}^{i1},i1}) \right\} ; 1 \le L_{RF}^{i1} \le N_{RTS} \quad (4.34)$$

$$RB_{i2} = \left\{ (u_{1,i2}, v_{1,i2}); (u_{2,i2}, v_{2,i2}); \ldots; (u_{L_{RB}^{i2},i2}, v_{L_{RB}^{i2},i2}) \right\} ; 1 \le L_{RF}^{i2} \le N_{RTS} \quad (4.35)$$

$$RB_{i3} = \left\{ (u_{1,i3}, v_{1,i3}); (u_{2,i3}, v_{2,i3}); \ldots; (u_{L_{RB}^{i3},i3}, v_{L_{RB}^{i3},i3}) \right\} ; 1 \le L_{RF}^{i3} \le N_{RTS} \quad (4.36)$$

Tal que:

$$(u_{B,i}, v_{B,i}) = (u_{1,i1}, v_{1,i1}) = (u_{1,i2}, v_{1,i2}) = (u_{1,i3}, v_{1,i3}); i = 1, 2, \ldots, N_{MB} \quad (4.37)$$

2. Calculamos la dirección para cada una de las rutas a través de la relación (4.33): $\theta_{B,i1}$, $\theta_{B,i2}$ y $\theta_{B,i3}$ y luego determinamos la distancia Euclidiana entre cada uno de los vectores unitarios vinculados a cada una de las orientaciones (ver sección 3.1.3.2.), es decir:

$$d_{E1} = 4\text{sen}^2\left(\frac{\theta_{B,i2} - \theta_{B,i3}}{2}\right) \quad (4.38)$$

$$d_{E2} = 4\text{sen}^2\left(\frac{\theta_{B,i3} - \theta_{B,i1}}{2}\right) \quad (4.39)$$

$$d_{E3} = 4\text{sen}^2\left(\frac{\theta_{B,i2} - \theta_{B,i1}}{2}\right) \tag{4.40}$$

3. El cálculo de la dirección se obtiene a través de la relación:

$$\theta_{B,i} = \begin{cases} \theta_{B,i1} & \text{si } d_{E1} = \min\{d_{E1}, d_{E2}, d_{E3}\} \\ \theta_{B,i2} & \text{si } d_{E2} = \min\{d_{E1}, d_{E2}, d_{E3}\} \\ \theta_{B,i3} & \text{si } d_{E3} = \min\{d_{E1}, d_{E2}, d_{E3}\} \end{cases} \tag{4.41}$$

En la figura 4.20 podemos apreciar la detección de las minucias de finalización y bifurcación, las orientaciones que presentan y las rutas que se han seguido a partir de las posiciones de las minucias, como establecimos al principio la información de importancia son el conjunto de tripletas MF y MB pero algunos algoritmos de coincidencia ([Xiping et al., 2000], [Ying et al, 2002]) también requieren de las rutas seguidas RF_i, RB_{i1}, RB_{i2} y RB_{i3}, sin embargo el inconveniente es el aumento del espacio en memoria requerido para almacenar esta información vinculada a una única impresión dactilar.

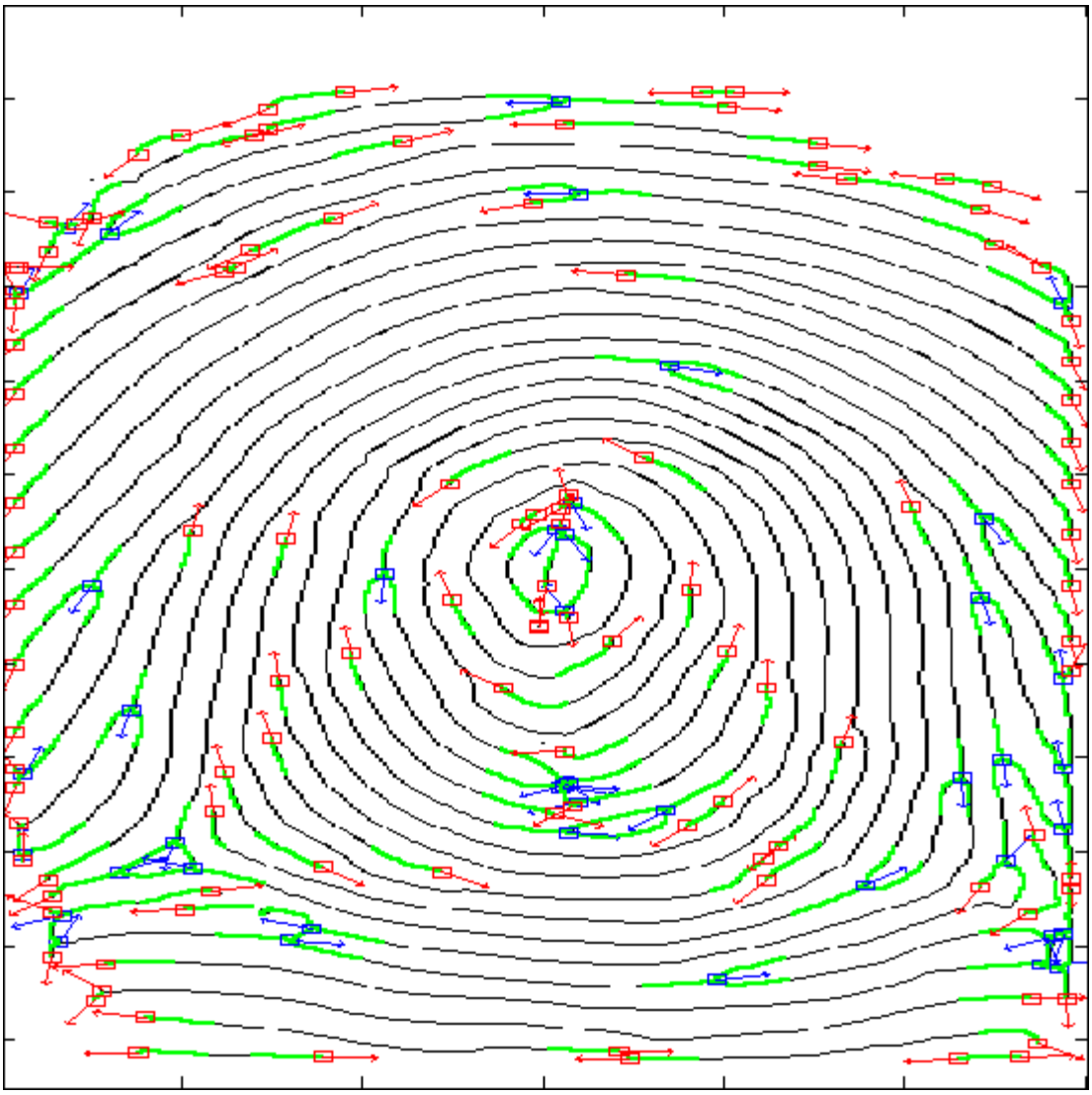


Figura 4.20 Detección de minucias de finalización (cuadrados rojos) y de bifurcación (cuadrados azules). Imagen 18_6.tif de la base de datos 2 del FVC2002.

4.3.1.3.3. Post procesamiento

Debido a las diferentes distorsiones que presenta toda impresión dactilar, en el paso de detección de las minucias un gran número de minucias falsas son detectadas y consideradas como válidas, diferenciar fiablemente las minucias falsas de las válidas es tarea del post procesamiento, siendo crucial para el reconocimiento automático ya que al eliminar la mayoría de minucias falsas se reduce significativamente el tiempo de coincidencia. En la figura 4.21 se muestran los tipos de minucias falsas.

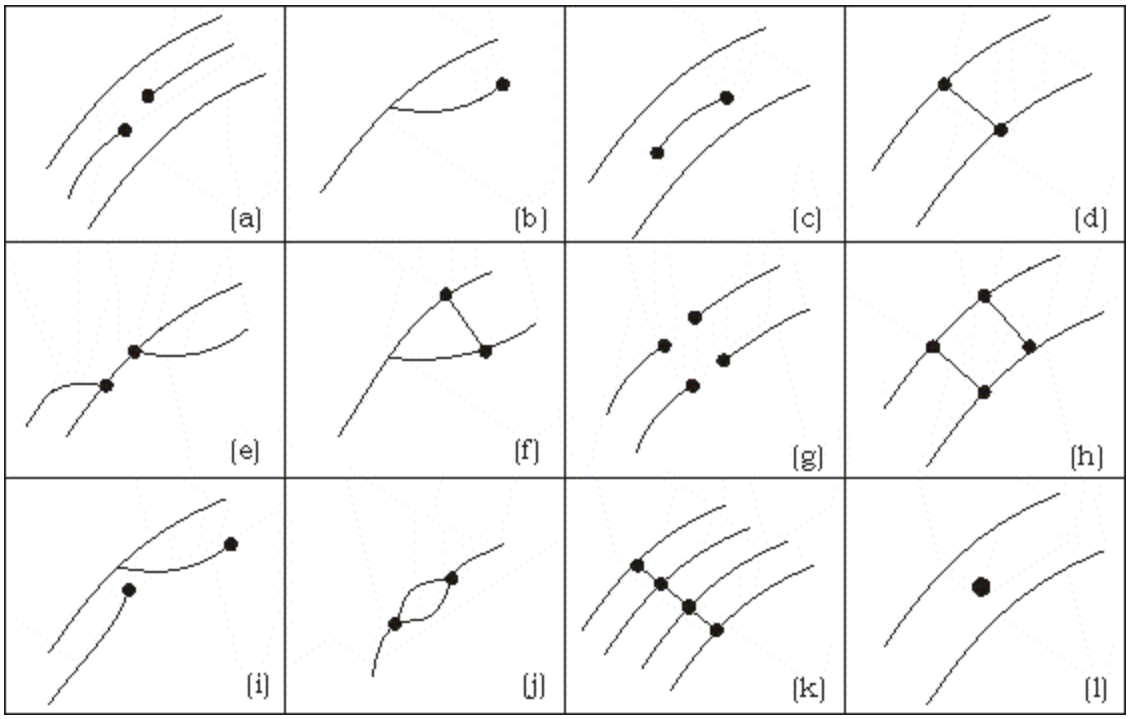


Figura 4.21 Tipos de minucias falsas: (a) cresta rota; (b) gancho; (c) isla; (d) puente; (e) *merge*; (f) triángulo; (g) múltiples crestas rotas; (h) *lader*; (i) cresta rota y *merge*; (j) lago; (k) *wrinkle*; (l) punto.

Puesto que cualquiera de los pasos de procesamiento particulares afectará el rendimiento de los pasos posteriores, se debe tener mucho cuidado en el orden de ejecución de los algoritmos para remover las minucias espurias, a continuación procedemos a explicar cada algoritmo siguiendo el mismo orden de ejecución.

Eliminación de ganchos: Para cada minucia de bifurcación determinamos si alguna de sus rutas termina en alguna minucia de finalización, si es así vemos la longitud que tenga dicha ruta, si ésta es menor que un valor umbral, entonces las correspondientes minucias de finalización y bifurcación son eliminadas del conjunto de tripletas MF y MB.

Eliminación de crestas rotas: Sobre cada minucia de finalización, se extrae su respectiva vecindad y se examina la posible presencia de otra minucia de finalización, si es así, comprobamos si la longitud entre las dos minucias es menor que un valor umbral, la diferencia angular de las direcciones de las minucias se encuentra entre 145º y 225º y las diferencias angulares entre la dirección de la

recta que une ambas minucias y la dirección de cada minucia se halla entre 0º y 40º, si todas estas condiciones se cumplen, entonces estamos frente a una cresta rota y ambas minucias son eliminadas del conjunto de tripletas MF.

Eliminación de islas: Para cada minucia de finalización cuya ruta termina en otra minucia de finalización, calculamos la longitud que tenga dicha ruta y si es menor que un valor umbral, entonces ambas minucias de finalización son eliminadas del conjunto de tripletas MF.

Eliminación de puentes: Para cada minucia de bifurcación, si alguna de sus rutas termina en otra minucia de bifurcación, se calcula la longitud que tenga dicha ruta y si es menor que un valor umbral, entonces ambas minucias de bifurcación son eliminadas del conjunto de tripletas MB.

Eliminación de minucias de borde: Sobre cada minucia de finalización y bifurcación se trazan radio vectores de longitud R y orientaciones de 0º, 45º, 90º, ..., 360º. Si el extremo de algún radio vector cae sobre una región no reconocible, entonces la minucia de finalización o bifurcación se elimina del conjunto de tripletas MF o MB.

En la figura 4.22 mostramos las minucias obtenidas después de la etapa de post procesamiento, puede apreciarse que la mayoría de minucias falsas han sido eliminadas.

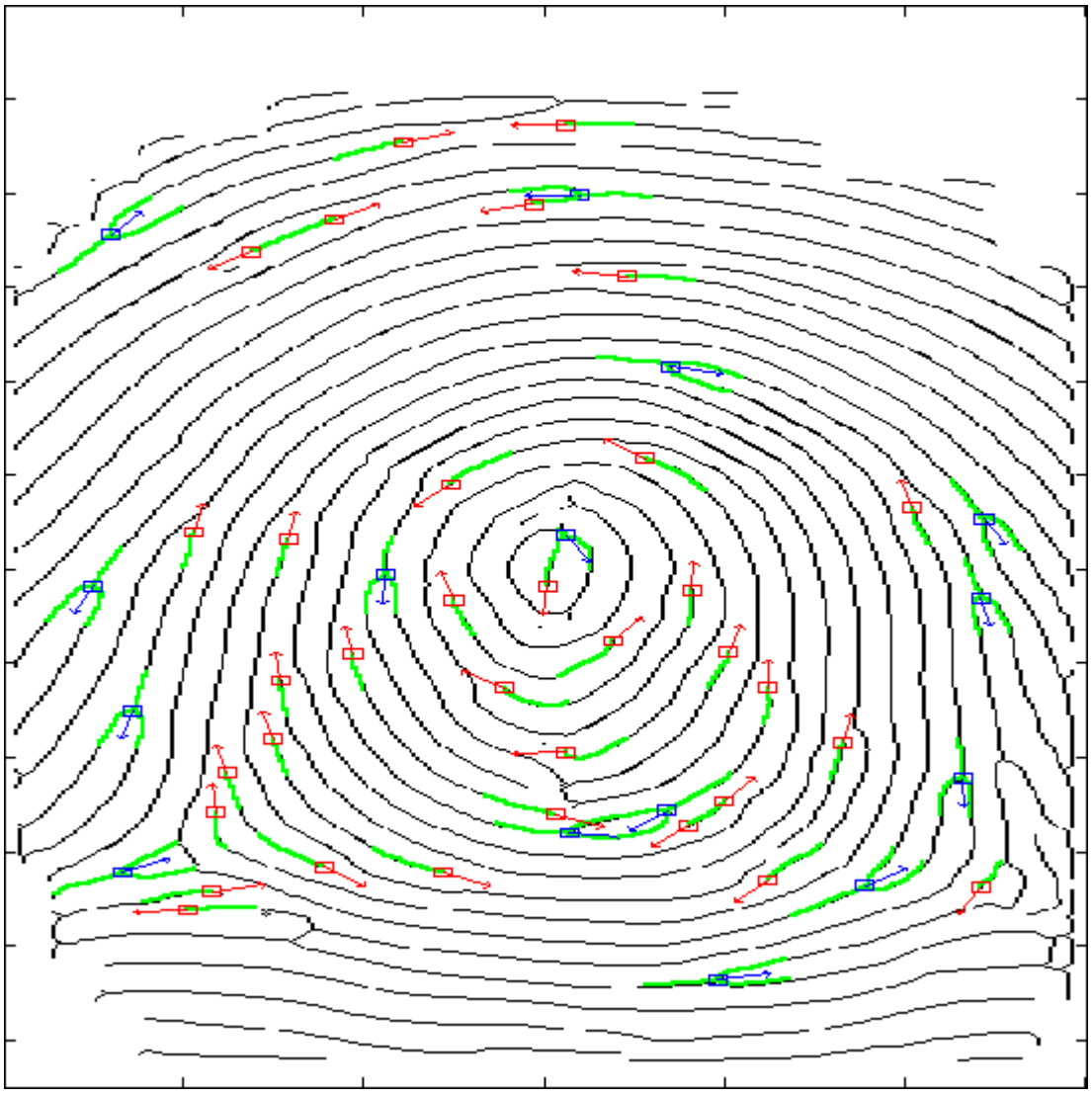


Figura 4.22 Minucias resultantes después de la etapa de post procesamiento. Imagen 18_6.tif de la base de datos 2 del FVC2002.

4.4. Extracción de características híbridas

Como se ha discutido en los párrafos anteriores, tradicionalmente son dos las principales características de las impresiones dactilares: las características globales y locales representadas por las minucias. Una impresión dactilar es clasificada usando el primer tipo de características e identificada/verificada usando el segundo tipo de características. No está claro que si el uso de estas características y filosofías fundamentales sean tan eficaces para los procesos de reconocimiento totalmente automáticos, ni sabemos si las tecnologías de identificación son capaces de tomar total ventaja de toda la información discriminatoria que puedan pro-

veer (En [Prabhakar, 2001, pp. 30-87] se estableció un límite superior para el rendimiento de los sistemas de coincidencia basados en las minucias), lo que implica la necesidad de explorar una representación alterna de las huellas dactilares que tengan un mejor poder de discriminación para sistemas automáticos motivada desde la perspectiva de la visión computacional y el reconocimiento de patrones. Esta representación complementaria debe combinar ambas fuentes de información, tanto locales como globales para no restringir el alcance de su aplicación. Una de tales representaciones fue planteada por [Jain et al., 1999b],[Jain et al., 1999c] y [Prabhakar, 2001] en la que se analizan las impresiones dactilares como patrones de texturas orientadas que exhiben un inherente y bien definido sentido de direccionalidad, argumentando que para texturas orientadas suficientemente complejas como las producidas por las huellas dactilares, pueden extraerse representaciones de textura invariantes combinando la información discriminatoria global y local.

El esquema de representación para texturas orientadas primero determina una plataforma de referencia en la impresión dactilar y define como región de interes a la región circular alrededor de dicha plataforma la cual se divide en sectores; cada sector se normaliza a una media y varianza constante. La región de interés normalizada es filtrada usando un banco de filtros de Gabor bidimensional obteniéndose un conjunto de imágenes filtradas; la varianza de los niveles de grises en un sector cuantifica la estructura de cresta subyacente y se define como una característica. Un vector de características denominado *Finger Code*, es la colección de todas las características vinculadas a cada una de las imágenes filtradas. De esta manera, la información local es representada por las características y la información global es representada por la enumeración ordenada en la sectorización.

4.4.1. Extracción del Finger Code

Los cinco pasos para la extracción del *Finger Code* son:

1. **Determinación del punto central**: Determinar el punto de referencia de la impresión dactilar.

2. **Sectorización circular**: Sectorizar la imagen en regiones circulares.

3. **Normalización**: Normalizar cada una de las regiones o sectores circulares a un determinado nivel de varianza y media constante.

4. **Filtrado**: Descomponer en el dominio espacio-frecuencia la imagen normalizada usando el banco de filtros de Gabor bidimensional, donde cada uno de los filtros está sintonizado a una orientación específica.

5. **Extracción del vector de características**: Calcular la varianza de los niveles de grises en los sectores circundantes alrededor del punto de referencia para cada una de las imágenes filtradas, obteniéndose el vector de características (*Finger Code*).

Veamos en detalle cada uno de estos pasos:

4.4.1.1. Determinación del punto de referencia

La detección del punto de referencia fue detallada en la sección 4.2.1., una condición no considerada y válida cuando se usa el *Finger Code* para la clasificación de huellas dactilares es que, después de detectar el punto de referencia este tiene que ser desplazado hacia abajo basado en el hecho de que la mayoría de la información categórica de una impresión dactilar yace en el Área Patrón (ver figura 4.23).



Figura 4.23 Desplazamiento del punto de referencia. Imagen 47_6.tif de la base de datos 2 del FVC2002.

4.4.1.2. Sectorización circular

Luego de obtener el punto de referencia de la imagen extraemos una región circular denominada región de interés, estando conformada por un número determinado de bandas. Cada banda está a su vez subdividida en un número de sectores $\mathbf{S}_i^{B,b,k}$ donde el sector i-ésimo es calculado en términos de los parámetros (r,θ), a través de:

$$\mathbf{S}_i^{B,b,k} = \{(u,v) \mid b.(T_i + 1) \le r < b.(T_i + 2), \theta_i \le \theta < \theta_{i+1}\} \quad (4.42)$$

Donde:

$$T_i = \lfloor i/k \rfloor \quad (4.43)$$

$$\theta_i = (i\%k)(2\pi/k) \quad (4.44)$$

$$r = \sqrt{(u-u_c)^2 + (v-v_c)^2} \quad (4.45)$$

$$\theta = \tan^{-1}((u-u_c)/(v-v_c)) \quad (4.46)$$

para todo: $i = \{0,....,B.k-1\}$

Siendo:

b: El ancho de cada banda.

k: El número de sectores considerados en cada banda.

B: El número de bandas concéntricas consideradas alrededor del punto de referencia para la extracción de características (ver figura 4.24).

El parámetro B depende del tamaño de la imagen y de la resolución (dpi) que presente. El parámetro b define el ancho de las bandas concéntricas y también depende de la resolución de la imagen. El parámetro k controla la captura de la información global versus la información local y depende de la aplicación.

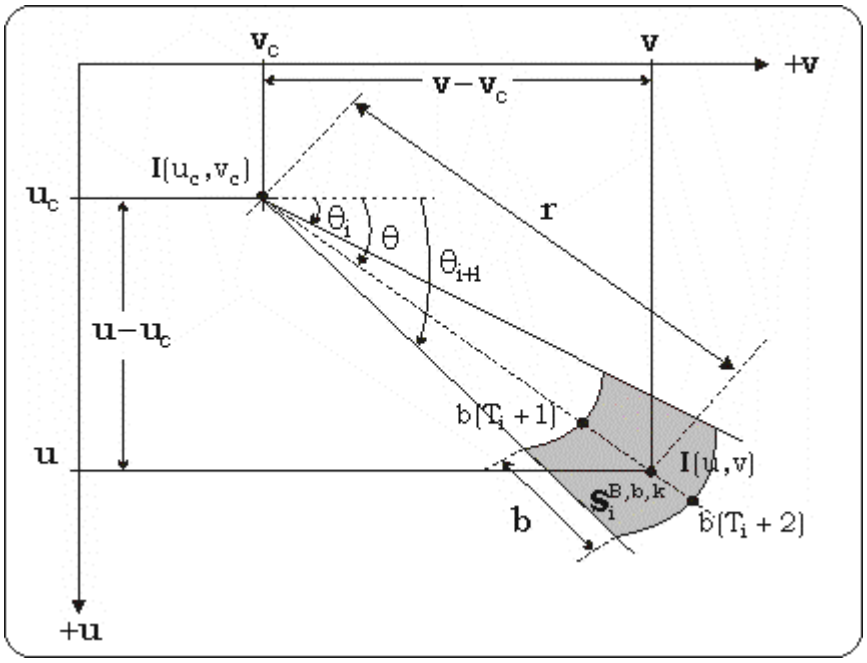


Figura 4.24 Parámetros del sector circular $\mathbf{S}_i^{B,b,k}$.

Para la implementación de la sectorización circular debemos percatarnos que la distribución de los sectores $\mathbf{S}_i^{B,b,k}$ está vinculada a una representación matricial siendo las bandas las filas y las columnas los sectores (ver figura 4.25). Se ha preferido usar la notación vectorial, donde el índice del vector viene dado por i, relacionándose con los índices de la matriz a través de las expresiones: $i = n.k + m$, tal que $m = i\%k$ y $n = \lfloor i/k \rfloor$. Es importante tomar en cuenta esta propiedad, ya que permite etiquetar cada píxel de la imagen dependiendo del sector al que pertenece, teniéndose como única información su magnitud y dirección a partir del punto de referencia. Para un píxel arbitrario (u, v) cuya dirección viene dada por θ y magnitud por r, se cumplirán las relaciones:

Para la dirección:

$$\theta_i \le \theta < \theta_{i+1} \Rightarrow i\%k\frac{2\pi}{k} \le \theta < (i+1)\%k\frac{2\pi}{k} \Rightarrow i\%k \le \frac{\theta.k}{2\pi} < (i+1)\%k \qquad (4.47)$$

Para la magnitud:

$$b(T_i+1) \le r < b(T_i+2) \Rightarrow (\left\lfloor \frac{i}{k} \right\rfloor + 1).b \le r < (\left\lfloor \frac{i}{k} \right\rfloor + 2).b \Rightarrow \left\lfloor \frac{i}{k} \right\rfloor \le \frac{r}{b} - 1 < \left\lfloor \frac{i}{k} \right\rfloor + 1 \quad (4.48)$$

De ambas desigualdades, aproximando al entero más cercano y tomando en cuenta las relaciones que definen a m, n e i podremos obtener la fila (banda) y la columna (sector) a la que pertenece el píxel (u,v).

$$m = i\%k = \lfloor \theta.k/2\pi \rfloor \tag{4.49}$$

$$n = \lfloor i/k \rfloor = \lfloor r/b - 1 \rfloor \tag{4.50}$$

$$i = \lfloor r/b - 1 \rfloor k + \lfloor \theta.k/2\pi \rfloor \tag{4.51}$$

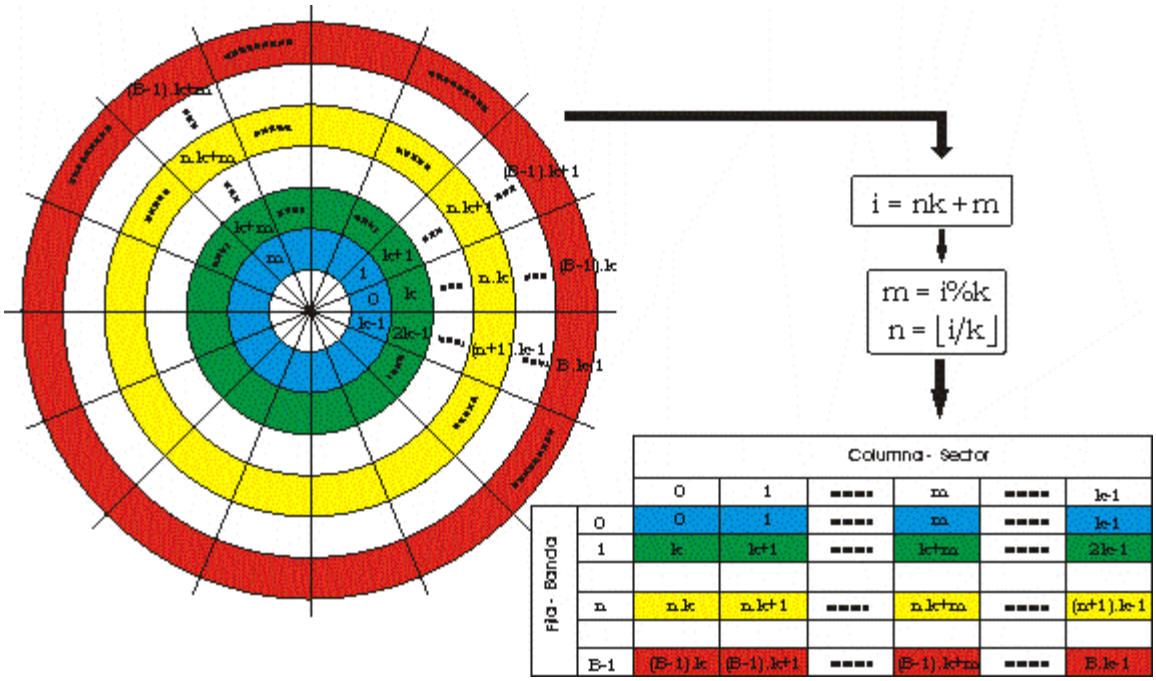


Figura 4.25 Obtención de la región de interés a través de una sectorización circular de la impresión dactilar centrado en su punto de referencia.

El valor de k debe ser escogido en base al intercambio entre la información local y global requerida para una aplicación particular, el valor de b debe ser escogido en base a la resolución de la imagen y la distancia promedio entre crestas. Una vez que los valores de los parámetros sean escogidos para una aplicación, éstos se definen como constantes. La banda más profunda (círculo) no es usada para la extracción de características debido a que los sectores en esta región están próximos al punto de referencia conteniendo muy pocos píxeles y por consiguiente la extracción de características en esta región no es muy fiable. En la figura 4.26

se muestra los pasos necesarios para la sectorización circular de la impresión dactilar.

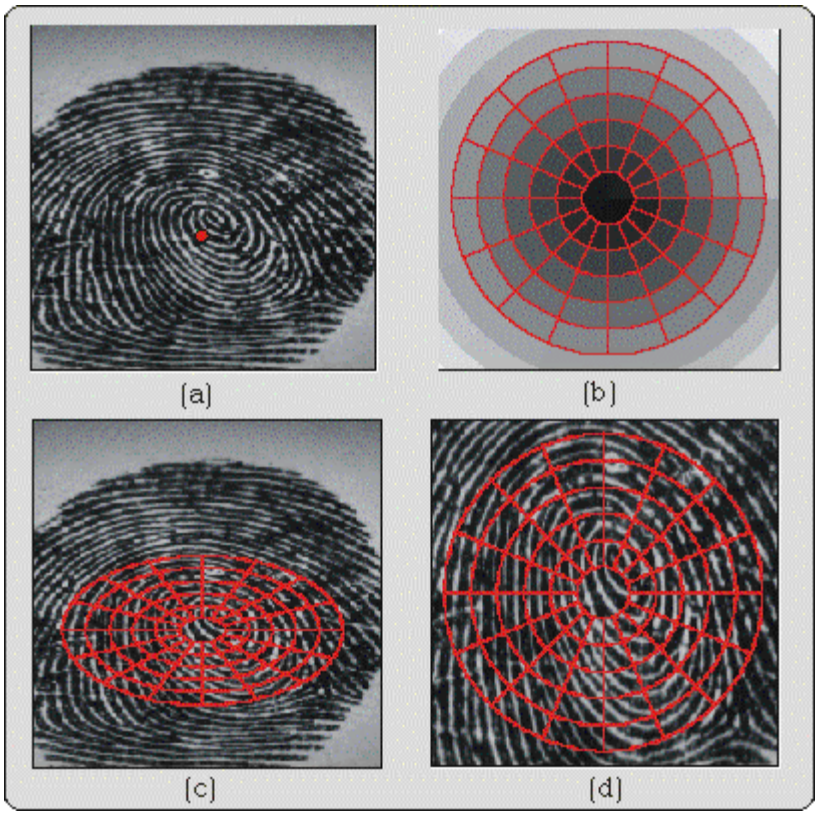


Figura 4.26 Obtención de la región de interés a través de una sectorización circular (a) punto de referencia; (b) imagen de regiones circulares; (c) imagen de regiones circulares superpuesta a la impresión dactilar centrada en el punto de referencia; (d) región rectangular que contiene a las regiones circulares. Imagen 47_6.tif de la base de datos 2 del FVC2002.

4.4.1.3. Normalización

Antes de filtrar la impresión dactilar, se normalizan las intensidades de los niveles de grises de cada sector a una varianza y media constante. La normalización es realizada para reducir los efectos del ruido y la distorsión no lineal debido a las diferentes presiones de la huella al ser capturada.

Si:

V_i y M_i : indican los valores de varianza y la media estimados para el sector

$\mathbf{S}_i^{B,b,k}$.

V_o y M_o: indican los valores de varianza y media deseados para cada sector.

Entonces el valor normalizado del píxel (u,v) perteneciente al sector $\mathbf{S}_i^{B,b,k}$ es definido a través de la relación:

$$\mathbf{NOR}_i(u,v) = \begin{cases} M_o + \sqrt{\frac{V_o}{V_i}}\,|\mathbf{I}(u,v) - M_i| & \mathbf{I}(u,v) > M_i \\ M_o - \sqrt{\frac{V_o}{V_i}}\,|\mathbf{I}(u,v) - M_i| & \text{en caso contrario} \end{cases} \qquad (4.52)$$

Al ser la normalización una operación realizada a nivel de píxeles, ésta no cambiará la claridad de las estructuras de crestas y valles (ver figura 4.27).

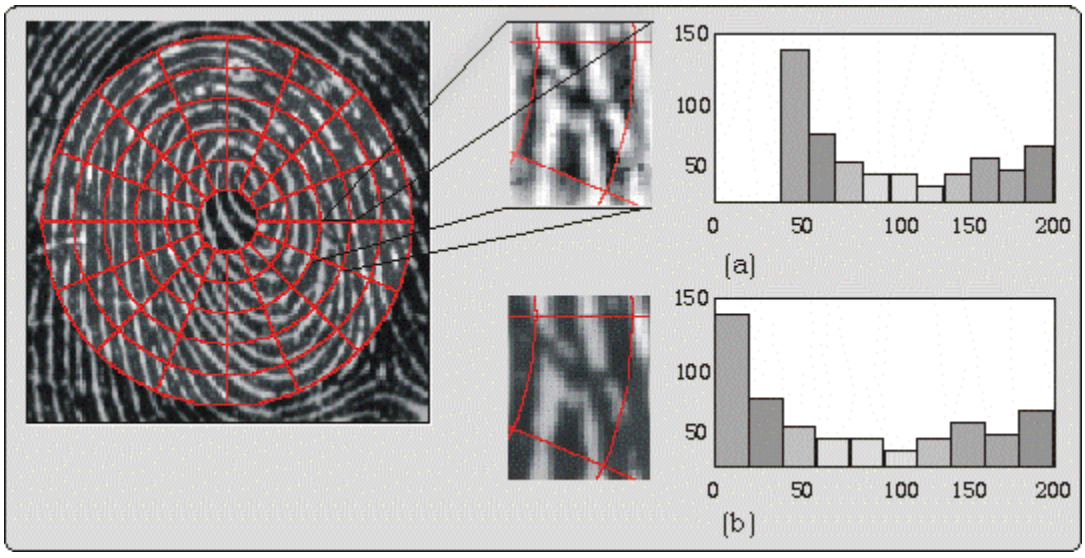


Figura 4.27 Normalización del sector circular (a) después de normalizar; (b) antes de normalizar. Imagen 47_6.tif de la base de datos 2 del FVC2002.

4.4.1.4. Filtrado

Para descomponer en el dominio del espacio-frecuencia las estructuras de crestas y valles filtramos la impresión dactilar en diferentes orientaciones usando un banco de filtros de Gabor bidimensional, que son filtros pasa banda con orientación y frecuencia selectiva (ver Anexo B). Para aplicar los filtros de Gabor

bidimensional a una imagen, se requiere especificar tres parámetros: la frecuencia espacial de la onda sinusoidal f, la orientación de los filtros θ y las desviaciones estándar de las envolventes Gaussianas δ_x y δ_y. La frecuencia espacial f es fijada al valor promedio de la frecuencia espacial de las crestas, $1/8$ píxel^{-1} (puesto que la distancia promedio entre crestas es aproximadamente ocho píxeles en una impresión dactilar tomada a 500 dpi). La orientación θ, determina el número de imágenes filtradas. Un filtro de Gabor bidimensional orientado a 0º acentúa las crestas que son paralelas al eje x y suaviza las crestas en otras orientaciones. Los filtros con diferentes orientaciones trabajan de una manera similar. Los valores para δ_x y δ_y son fijados a 4 píxeles (sobre la mitad del promedio de la distancia entre crestas.). El filtraje se realiza en el dominio espacial con una máscara de tamaño de 33x33 píxeles (ver figura 4.28). La convolución (ver Anexo B) con las máscaras de Gabor es el paso que mayor tiempo consume al momento de la obtención del *Finger Code*.

4.4.1.5. Extracción del vector de características

La característica vinculada a cada uno de los sectores para cada imagen filtrada se define como la desviación estándar de cada sector, es decir:

$$V_{i\theta} = \sqrt{\sum_{u=u_1}^{u_{K_i}} \sum_{v=v_1}^{v_{K_i}} (\mathbf{F}_{i\theta}(u,v) - \mu_{i\theta})^2} \qquad (4.53)$$

Donde:

$V_{i\theta}$: Es el valor de la característica i-ésima vinculada a la imagen filtrada en la orientación θ, para todo $i = \{0,1,\cdots,79\}$

$\mathbf{F}_{i\theta}(u,v)$: Es el píxel en la posición (u,v) de la imagen filtrada en la orientación θ perteneciente al sector $\mathbf{S}_i^{B,b,k}$.

K_i: Es el número de píxeles en $\mathbf{S}_i^{B,b,k}$

$\mu_{i\theta}$: Es el valor promedio de los píxeles de $\mathbf{F}_{i\theta}$.

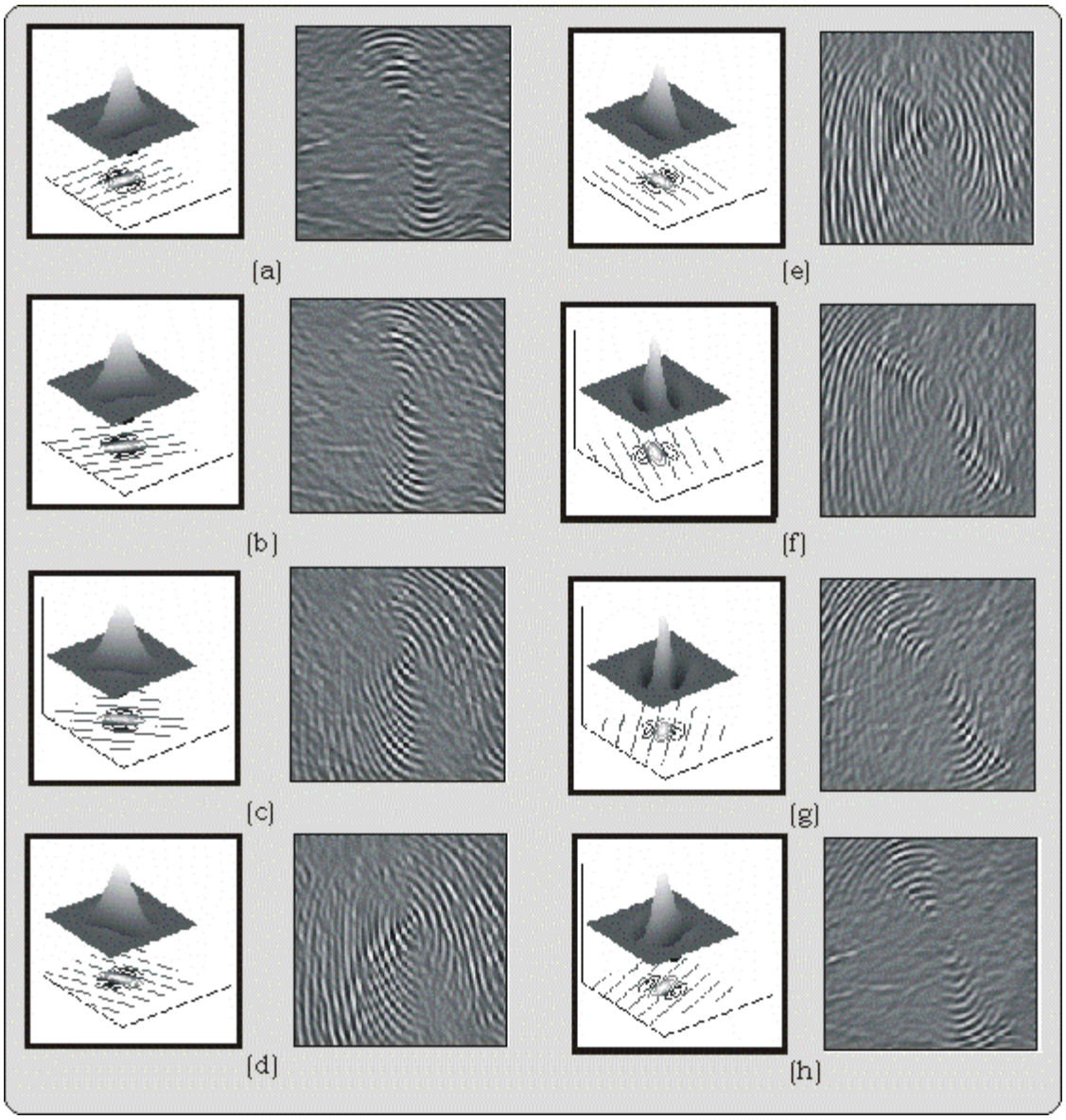


Figura 4.28. Imágenes filtradas con el banco de filtros de Gabor bidimensional para diferentes orientaciones (a)0º ; (b)22.5º; (c)45º; (d)67.5º; (e)90º; (f)112.5º; (g)135º; (h)157.5º . Imagen 47_6.tif de la base de datos 2 del FVC2002.

$V_{i\theta}$ define los elementos del vector de características de la impresión dactilar, denominado *Finger Code* (semejante al *Iris Code* introducido por [Daugman, 1993]). El número de orientaciones para los filtros viene especificado por la aplicación; por ejemplo, para la verificación se utilizan ocho filtros orientados y para la clasificación se usan cuatro filtros orientados. Como está indicado en [Jain et al., 2000] un aumento o disminución en el número de filtros no provee un incremento en el rendimiento de la verificación o clasificación de huellas dactilares.

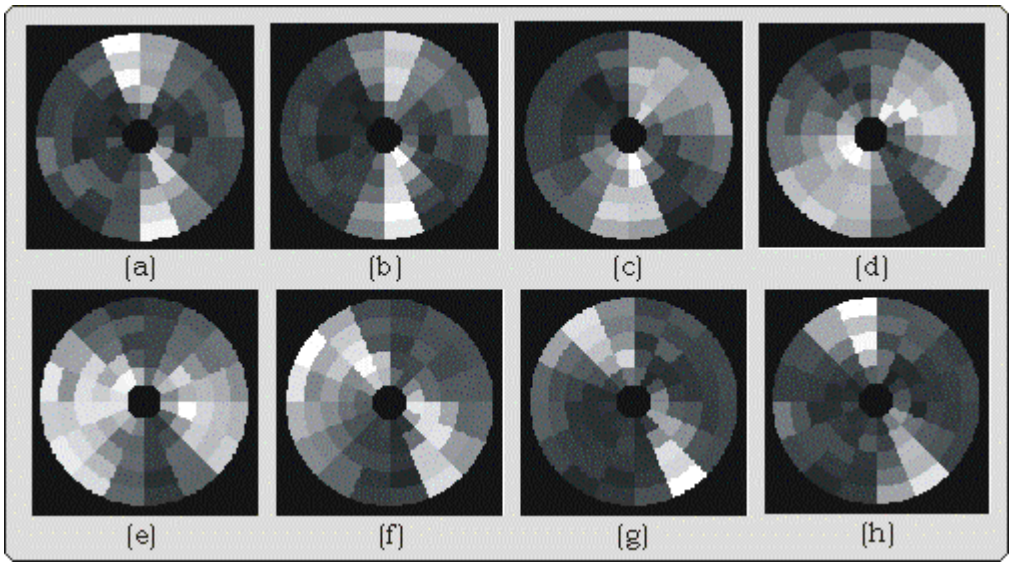


Figura 4.29 (a)0º ; (b)22.5º; (c)45º; (d)67.5º; (e)90º; (f)112.5º; (g)135º; (h)157.5º. Imagen 47_6.tif de la base de datos 2 del FVC2002.

4.5. Resultados Experimentales

4.5.1. En la etapa de extracción de características globales

En este experimento utilizamos la base de datos 2 del FVC2002, de ella seleccionamos ciertas imágenes de manera que presenten diversos niveles de calidad y a su vez formen un conjunto representativo de las cinco clases básicas de impresiones dactilares (en total se han seleccionado 280 imágenes). Se ha considerado el tamaño de la ventana para la sectorizaciòn rectangular de 35x37 píxeles.

El rendimiento de los algoritmos para detectar los puntos singulares de una impresión dactilar se han analizado considerando únicamente la capacidad que tienen éstos de detectar con precisión el punto de referencia, puesto que su detección es de mayor relevancia para el proceso de inscripción de una impresión y para el cálculo del *Finger Code*. Visto de esta manera, el análisis de la robustez y precisión de los algoritmos queda resumido al determinar la influencia que tiene la calidad y la rotación de la imagen en la localización del punto de referencia. Se

ha desarrollado un conjunto de pruebas que permiten analizar esta dependencia, las cuales pasaremos a describir brevemente.

4.5.1.1. Dependencia del punto de referencia respecto a la calidad de la imagen

Para validar el desempeño de los algoritmos considerando la calidad de la imagen realizamos los siguientes pasos:

1. Determinamos en forma manual la localización del punto de referencia para una imagen seleccionada.
2. Determinamos el índice de calidad que presenta la imagen y la localización del punto de referencia a través del algoritmo en cuestión.
3. Determinamos la distancia real en milímetros (=0.0446x$Error_{(píxel)}$) entre los puntos de referencia determinados en (1) y (2), el resultado es clasificado dependiendo de la calidad que presente la imagen.
4. Los pasos (1), (2) y (3) son ejecutados hasta considerar todas las imágenes seleccionadas de la base de datos.

De las figuras 4.30 y 4.31 podemos inferir que el algoritmo basado en los componentes sinusoidales posee un mejor desempeño puesto que presenta un error promedio de 1.2±0.8 milímetros para imágenes de pobre calidad frente a los 1.3±1.8 milímetros de error promedio que presenta el algoritmo basado en el índice de Poincaré. Los errores promedios disminuyen a 1.1±0.6 y 0.7±0.5 milímetros respectivamente para imágenes de buena calidad (ver figura 4.30).

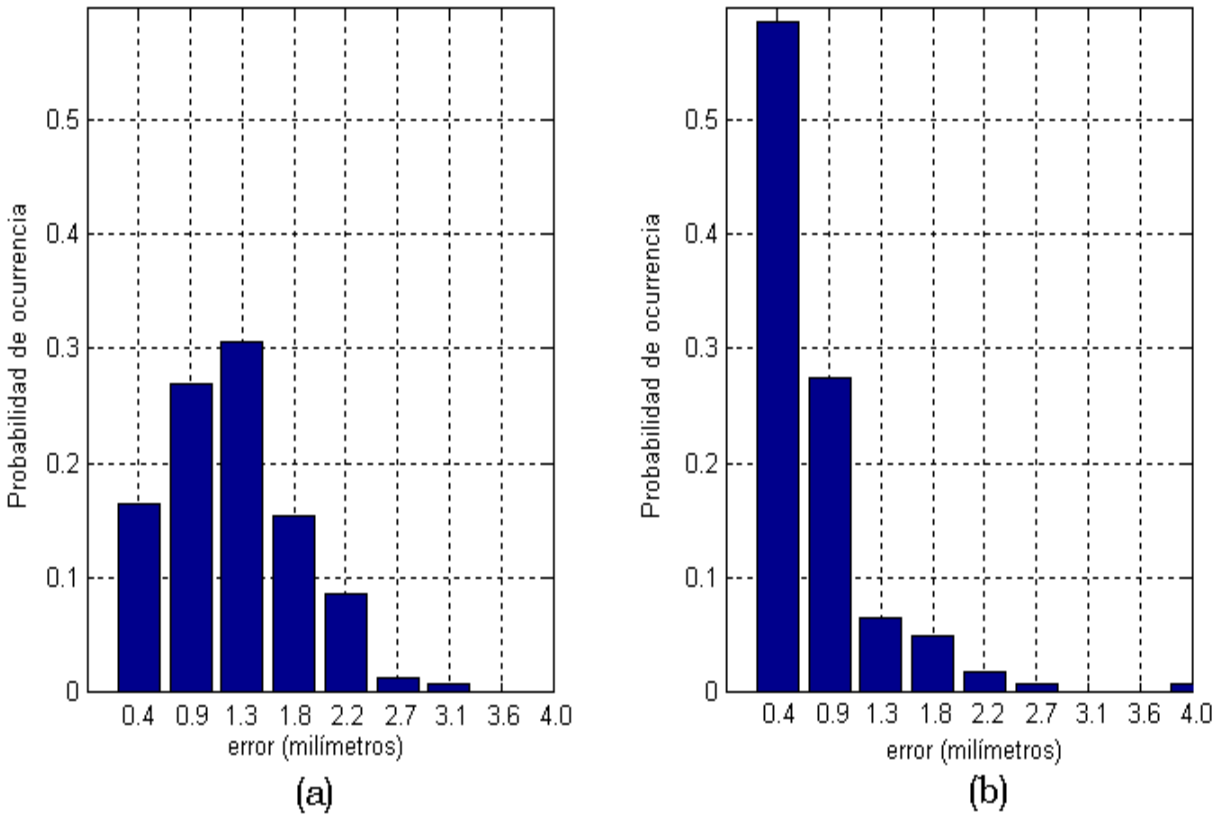


Figura 4.30 Histogramas de error en la determinación del punto de referencia para imágenes de buena calidad (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

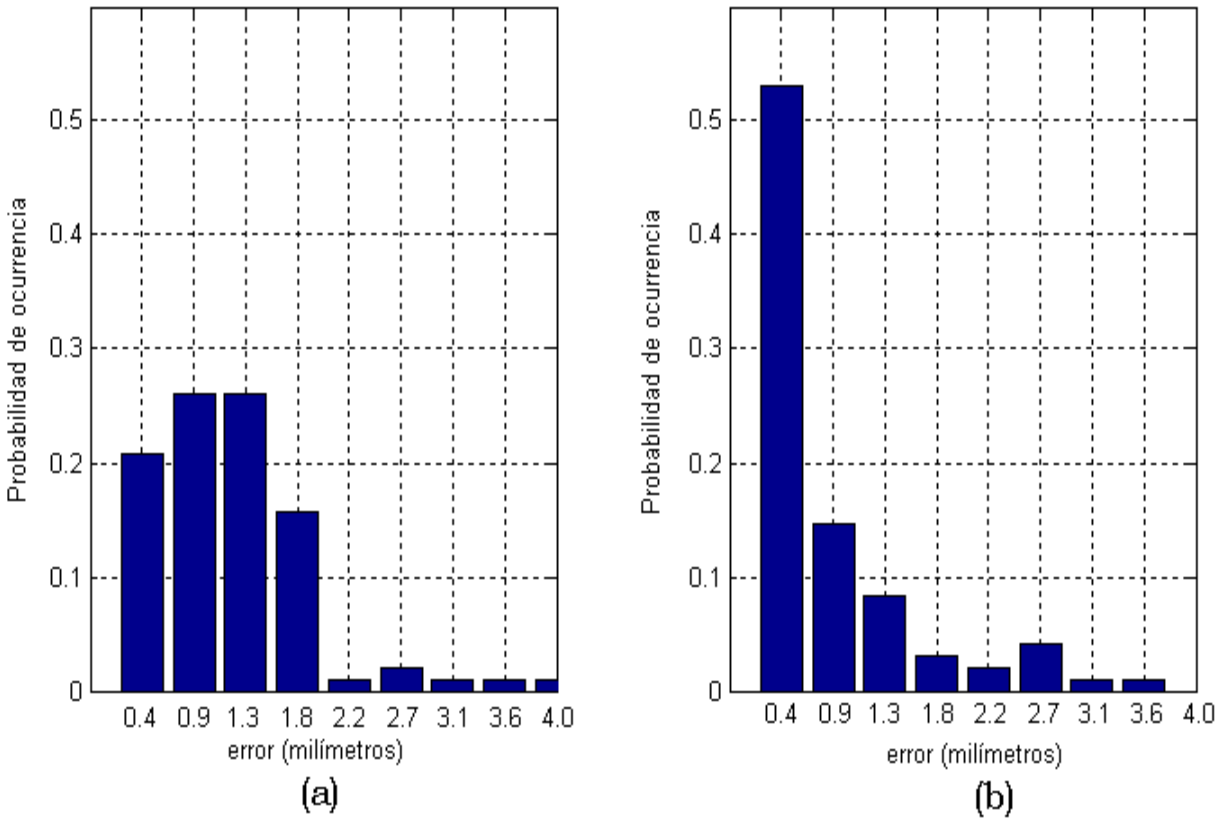


Figura 4.31 Histogramas de error en la determinación del punto de referencia para imágenes de pobre calidad (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

4.5.1.2. Dependencia del punto de referencia respecto a la rotación de la imagen

Para validar el desempeño de los algoritmos frente a la rotación que presenta una impresión dactilar es necesario contar con impresiones rotadas en intervalos regulares y constantes, puesto que no es posible adquirir impresiones con estas características, usamos la función *imrotate* de MATLAB para generarlas; el intervalo de rotación considerado es de -45º a +45º con espaciamientos de 2.5º. No consideramos rotaciones mayores porque implicaría que el dedo se encuentra fuera del área de captura del sensor. Esta validación se realiza a través de los siguientes pasos:

1. Determinamos en forma manual la localización del punto de referenci para una imagen seleccionada.
2. Rotamos la imagen y localizamos el punto de referencia a través del algoritmo en cuestión.
3. El punto determinado en (2) es llevado al sistema coordenado establecido para la imagen inicial.
4. Determinamos la distancia real entre los puntos de referencia determinados en (1) y (3), ésta se da en milímetros (ver sección anterior).
5. Los pasos (2), (3) y (4) son ejecutados hasta considerar todas las muestras en el intervalo de rotación.

Las figuras 4.32, 4.33, 4.34, 4.35 y 4.36 muestran los resultados de esta prueba efectuada en impresiones dactilares de diferentes clases, considerando las imágenes rotadas en el intervalo referido anteriormente. Concluimos que para la mayoría de clases el comportamiento del algoritmo basado en el índice de Poincaré es más robusto y preciso ante la variación en la rotación de la imagen, teniendo como excepción las huellas de tipo A, donde son evidentes las ventajas que presenta la técnica basada en las derivadas de los componentes sinusoidales (ver figura 4.34). Además podemos ver que los errores son significativos por encima de los 30º lo que no es preocupante, puesto que las rotaciones involuntarias generadas por los usuarios son inferiores a 30º.

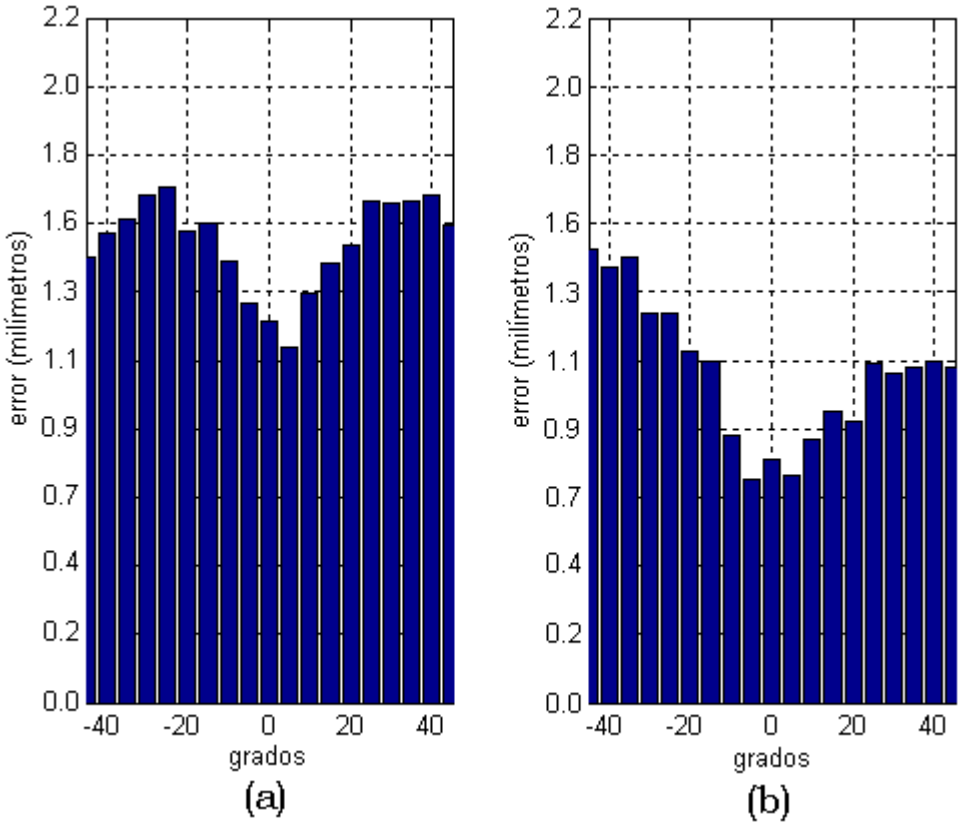


Figura 4.32 Gráfica del error en la determinación del punto de referencia para impresiones dactilares tipo *Left loop* (LL) a medida que la imagen es rotada (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

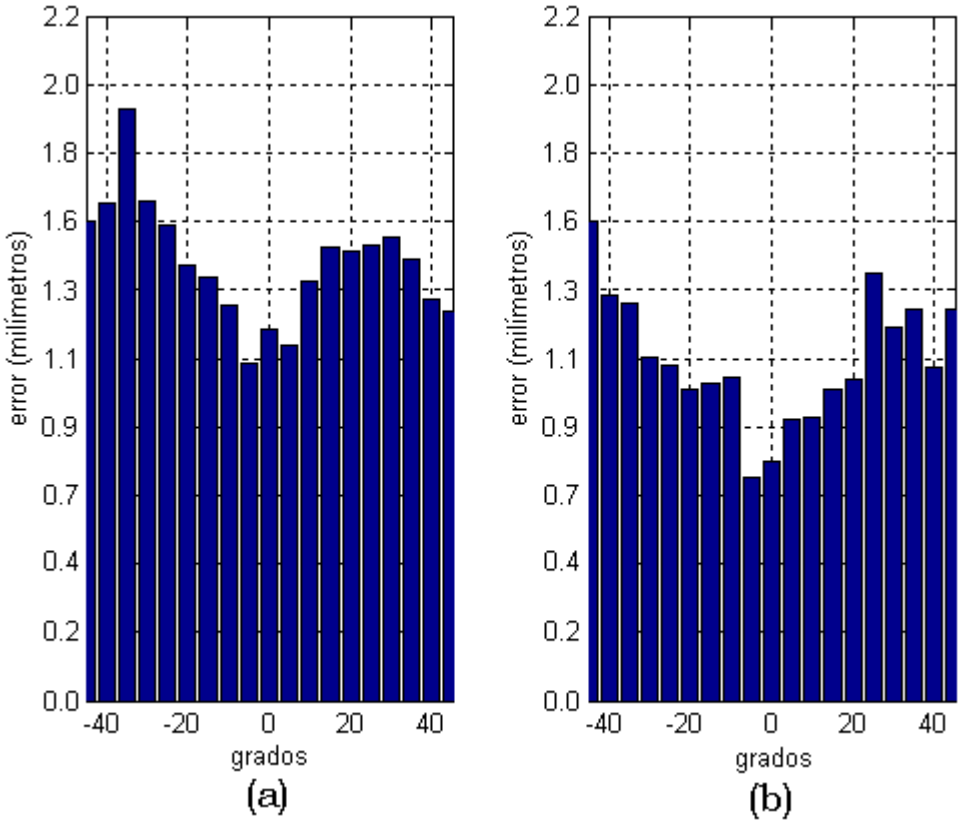


Figura 4.33 Gráfica del error en la determinación del punto de referencia para impresiones dactilares tipo *Right loop* (RL) a medida que la imagen es rotada (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

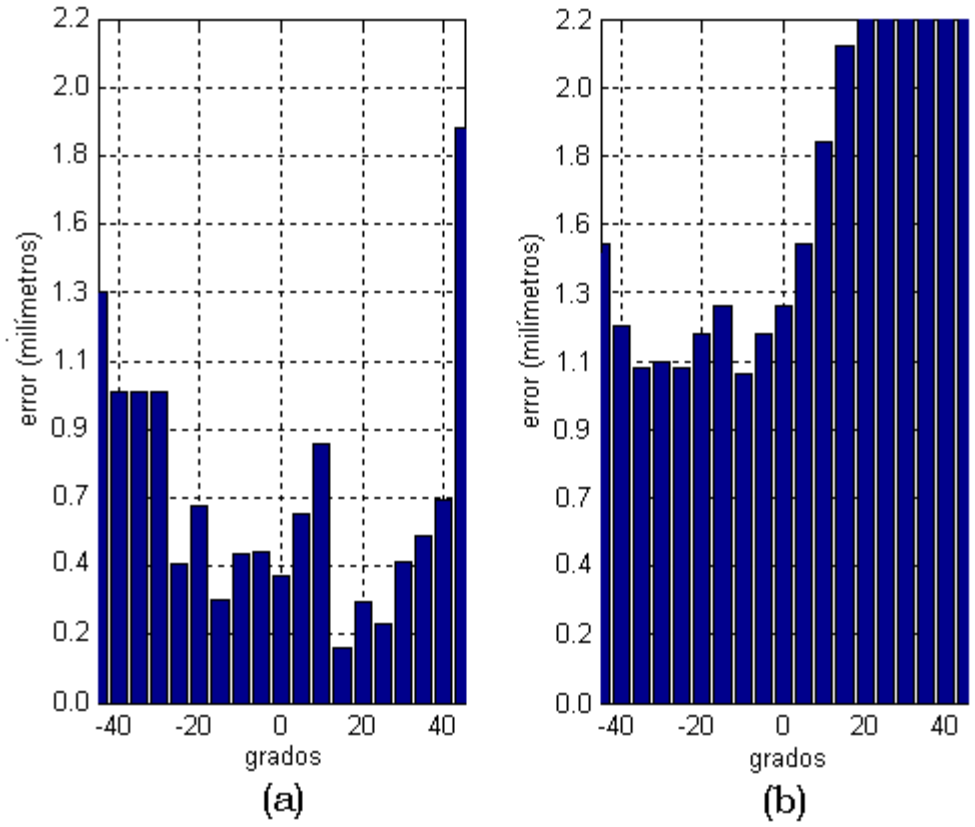


Figura 4.34 Gráfica del error en la determinación del punto de referencia para impresiones dactilares tipo *Arch* (A) a medida que la imagen es rotada (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

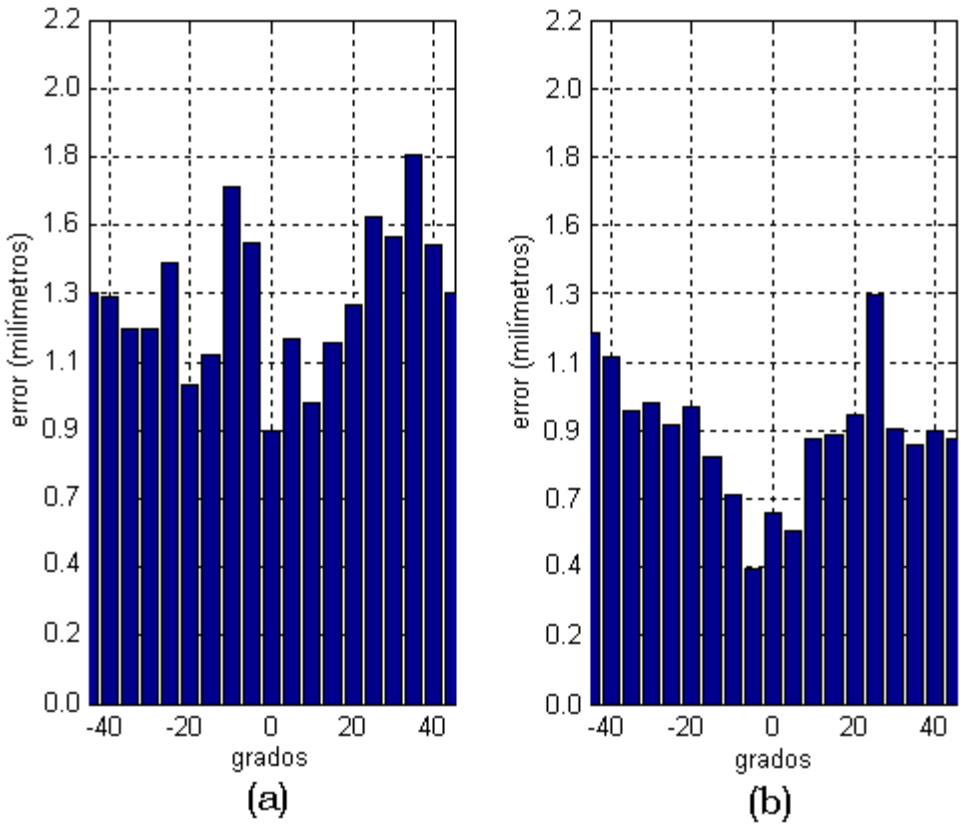


Figura 4.35 Gráfica del error en la determinación del punto de referencia para impresiones dactilares tipo *Whorl* (W) a medida que la imagen es rotada (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

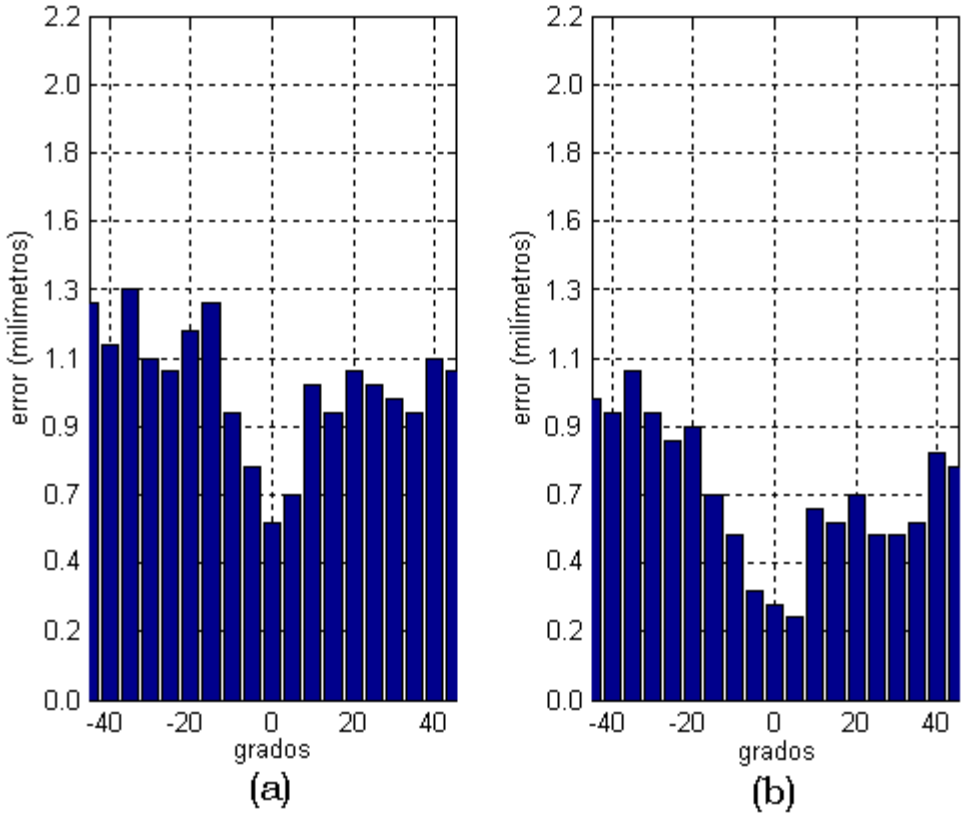


Figura 4.36 Gráfica del error en la determinación del punto de referencia para impresiones dactilares tipo *Tented Arch* (TA) a medida que la imagen es rotada (a) técnica basada en las componentes sinusoidales de la Imagen de Orientación; (b) técnica basada en el índice de Poincaré.

4.5.2. En la etapa de extracción de características locales

El rendimiento de la técnica para la extracción de minucias se evalúa por la comparación de las minucias de una imagen obtenidas por la técnica en cuestión versus el conjunto de minucias obtenidas por un experto. Aunque este es un proceso laborioso, provee una medida de evaluación del rendimiento imparcial denominada Índice de Virtud [Hong et al., 1998].

Para el cálculo del Índice de Virtud es necesario tomar en cuenta las siguientes definiciones: Sea $\upsilon_A = \{m_1^A, m_2^A, \ldots, m_{N_A}^A\}$ el conjunto de N_A minucias obtenidas por la técnica propuesta y $\upsilon_R = \{m_1^R, m_2^R, \ldots, m_{N_R}^R\}$ el conjunto de N_R minucias obtenidas por un experto para una misma imagen, entonces:

Minucias apareadas: Una minucia detectada por el algoritmo m_i^A, y una minucia detectada por el experto m_j^R, se dice que son apareadas si m_i^A se encuentra en el interior de un cuadro de tolerancia de tamaño 9x9 píxeles centrado en m_j^R (ver figura 4.37.a).

Minucia perdida: Aquella minucia que será insertada en el conjunto υ_A (no detectada por el algoritmo), para poder aparearla con alguna minucia del conjunto υ_R (ver figura 4.37.b).

Minucia espuria: Aquella minucia del conjunto υ_A que requiere ser eliminada debido a que no puede ser apareada con ninguna minucia del conjunto υ_R (ver figura 4.37.c).

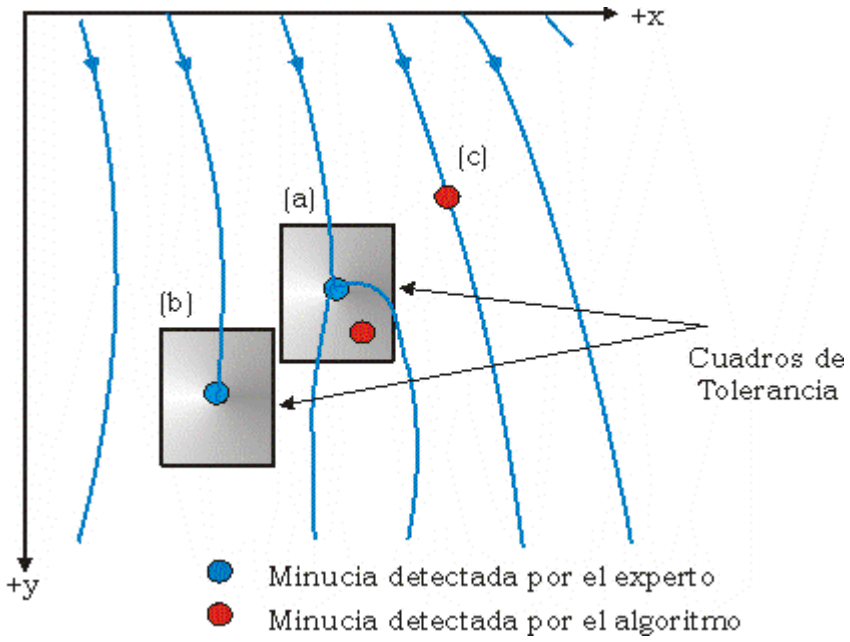


Figura 4.37 (a) Minucia apareada; (b) minucia perdida; (c) minucia destruida.

El Índice de Virtud es definido a través de la siguiente relación:

$$GI = \frac{\sum_{i=0}^{N_{Fil}-1}\sum_{j=0}^{N_{Col}-1} Cld_{(i,j)}.(Aprd_{(i,j)} - Dstr_{(i,j)} - Intr_{(i,j)})}{\sum_{i=0}^{N_{Fil}-1}\sum_{j=0}^{N_{Col}-1} Cld_{(i,j)}.Rls_{(i,j)}} \qquad (4.54)$$

Donde:

$Aprd_{(i,j)}$: Es el número de minucias apareadas en la r-ésima ventana.

$Cld_{(i,j)}$: Es el factor de calidad del bloque (i, j) (bueno = 4, medio = 3, pobre = 2, segundo plano = 1).

$Dstr_{(i,j)}$: Es el número de minucias destruidas en el bloque (i, j).

$Intr_{(i,j)}$: Es el número de minucias insertadas en el bloque (i, j).

$Rls_{(i,j)}$: Es el número de minucias reales en el bloque (i, j).

Definimos las variables $Aprd$, $Dstr$, $Intr$, y Rls como sigue:

$$Aprd = \sum_{i=0}^{N_{Fil}-1}\sum_{j=0}^{N_{Col}-1} Aprd_{(i,j)} \qquad (4.55)$$

$$Dstr = \sum_{i=0}^{N_{Fil}-1} \sum_{j=0}^{N_{Col}-1} Dstr_{(i,j)} \tag{4.56}$$

$$Intr = \sum_{i=0}^{N_{Fil}-1} \sum_{j=0}^{N_{Col}-1} Intr_{(i,j)} \tag{4.57}$$

$$Rls = \sum_{i=0}^{N_{Fil}-1} \sum_{j=0}^{N_{Col}-1} Rls_{(i,j)} \tag{4.58}$$

Los pasos para la determinación del Índice de Virtud son:

1. Determinamos los conjuntos υ_A (minucias detectada por el experto) y υ_R (minucias detectada por el algoritmo).
2. Sectorizamos la imagen rectangularmente (ver sección 3.2.1.1.) y para cada bloque (i, j) determinamos su respectivo índice de calidad $Cld_{(i,j)}$ (ver sección 3.4.1.2.2.) y los números de correspondencias entre minucias $Aprd_{(i,j)}$, $Dstr_{(i,j)}$, $Intr_{(i,j)}$ y $Rls_{(i,j)}$.
3. A través de la relación (4.54) determinamos el respectivo Índice de Virtud GI.
4. A través de las relaciones (4.55), (4.56), (4.57) y (4.58) determinamos las medidas de ponderación $Aprd$, $Dstr$, $Intr$ y Rls.

En la figura 4.38 se muestran las minucias detectadas por el experto y por el algoritmo (figuras 4.38.a y 4.38.b) y los bloques vinculados a las posiciones de las minucias reales, apareadas, insertadas y eliminadas.

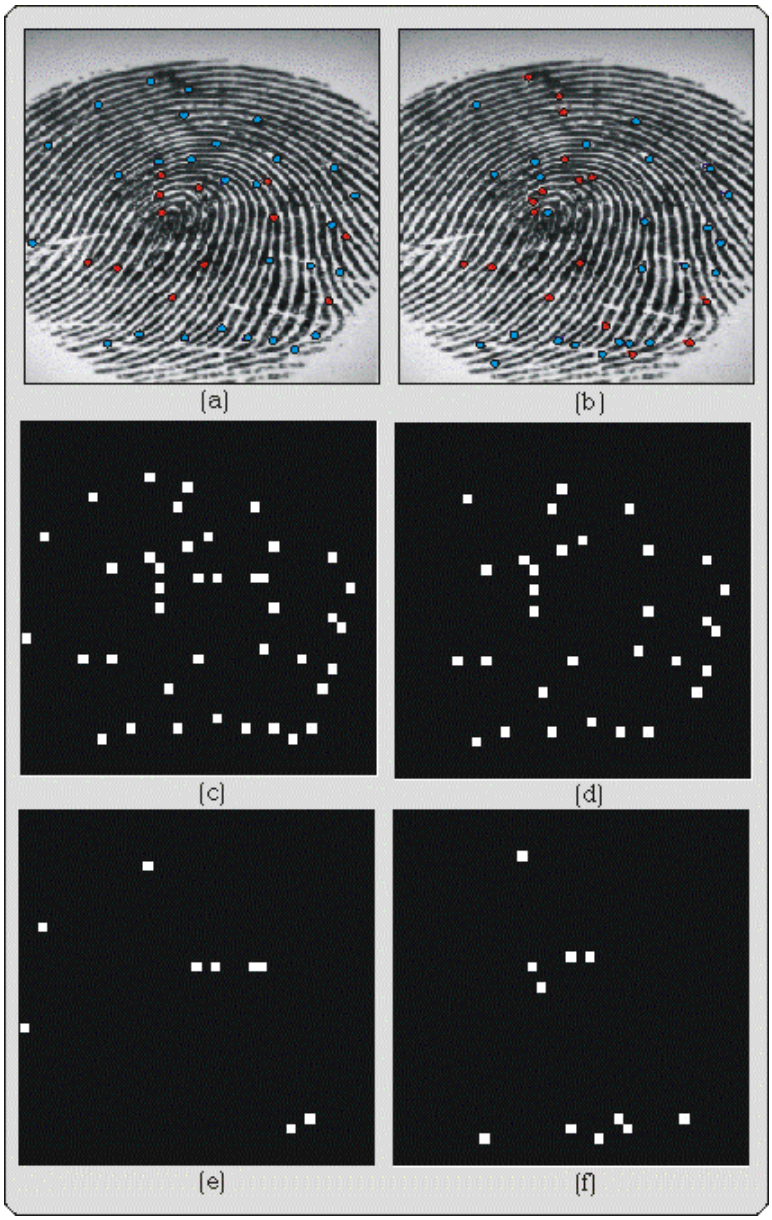


Figura 4.38 (a) Minucias detectadas por el experto; (b) minucias detectadas por el algoritmo; (c) bloques vinculados a posiciones de minucias reales; (d) bloques vinculados a posiciones donde las minucias están apareadas; (e) bloques vinculados a posiciones donde las minucias son insertadas; (f) bloques vinculados a posiciones donde las minucias son destruidas. Imagen 14_2.tif de la base de datos 2 del FVC2002.

El máximo valor de GI es +1 el cual es obtenido cuando $Dstr_{(i,j)} = Intr_{(i,j)} = 0$ y $Aprd_{(i,j)} = Rls_{(i,j)}$ para todos los pares de minucias detectadas. Note que el número de coincidencias en cada ventana es ponderado por la calidad. Por lo tanto, un

error (minucias espurias o perdidas) en una ventana de buen contraste es ponderado más fuerte comparado con el mismo error en un área ruidosa de la imagen. Si asumimos que el número total de minucias detectadas en una ventana es a lo sumo dos veces el número de minucias reales, entonces el mínimo valor de GI es -3. Este valor es obtenido cuando $Aprd_{(i,j)} = 0$, $Dstr_{(i,j)} = 2.Rls_{(i,j)}$ y $Intr_{(i,j)} = Rls_{(i,j)}$.

Un valor positivo de GI implica que el algoritmo de extracción de características realiza un buen trabajo sobre la imagen. Los valores GI para un subconjunto representativo de diez impresiones dactilares de la base de datos 2 del FVC2002 son mostrado en la tabla 4.2. El valor promedio de GI usando la técnica de extracción de características propuesto es 0.30446. El máximo y mínimo valor de GI obtenido para este conjunto de datos son 0.45968 y 0.11348, respectivamente.

Para poder determinar el significado de los valores observados, los comparamos versus los valores de GI obtenidos bajo una distribución *Baseline*. Hemos usado el siguiente procedimiento para obtener la distribución *Baseline* de GI.

1. Generamos un par de números enteros (x,y) tal que x e y pertenecen al dominio [0,559]x[0,295] (tomar en cuenta que las imágenes son de tamaño 560x296).
2. Si (x,y) es un punto perteneciente al primer plano en la imagen de entrada, entonces aceptamos el punto como un punto de minucia aleatorio.
3. Si el número total de puntos de minucia aleatorios es igual al número de puntos minucias obtenidos desde el algoritmo de extracción el proceso finaliza; en caso contrario volvemos al paso 1.

IMAGEN	GI	Rls	Aprd	Intr	Dstr	GI^P	$Aprd^P$	$Intr^P$	$Dstr^P$
14_2.tif	0.45968	40	31	9	11	-1.7097	4	36	38
7_2.tif	0.44444	32	28	4	12	-1.812	3	29	37
13_1.tif	0.42697	24	20	4	5	-1.7191	2	22	23
1_5.tif	0.40789	20	18	2	8	-1.8684	1	19	25
4_2.tif	0.36066	19	18	1	16	-2.4262	2	17	32
3_5.tif	0.34066	24	20	4	8	-1.7253	2	22	26
5_2.tif	0.19512	23	18	5	8	-1.7561	2	21	24
15_2.tif	0.16374	47	33	14	11	-1.5322	5	42	39
10_2.tif	0.13194	36	24	12	8	-1.5556	3	33	29
11_3.tif	0.11348	44	30	14	11	-1.6383	4	40	37
PROMEDIOS	0.30446					-1.7743			

Tabla 4.2 Valores de GI para diez impresiones dactilares de la base de datos 2 del FVC2002.

El valor de GI calculado con un conjunto de N_A puntos aleatorios con los N_R puntos reales puede ser usado para evaluar el valor observado de GI. La distribución de *Baseline* para la impresión 14_2.tif (basado sobre 100 conjuntos de números aleatorios) es mostrada en la figura 4.39. Para las diez impresiones usadas en la tabla 4.2, también se proveen los valores de *Baseline* GI (denotadas por GI^P). Note que las variables $Aprd^P$, $Dstr^P$, $Intr^P$ y Rls^P corresponden a Aprd, Dstr, Intr y Rls respectivamente, para la distribución *Baseline*. El valor de GI^P varía de -2.4262 a -1.5556 con un valor promedio de -1.7743. (Note que el valor promedio de GI es 0.30446). Comparando los valores de GI^P con los valores de GI calculado desde las minucias extraídas, podemos concluir que la técnica de extracción de minucias propuesta es robusta y exacta.

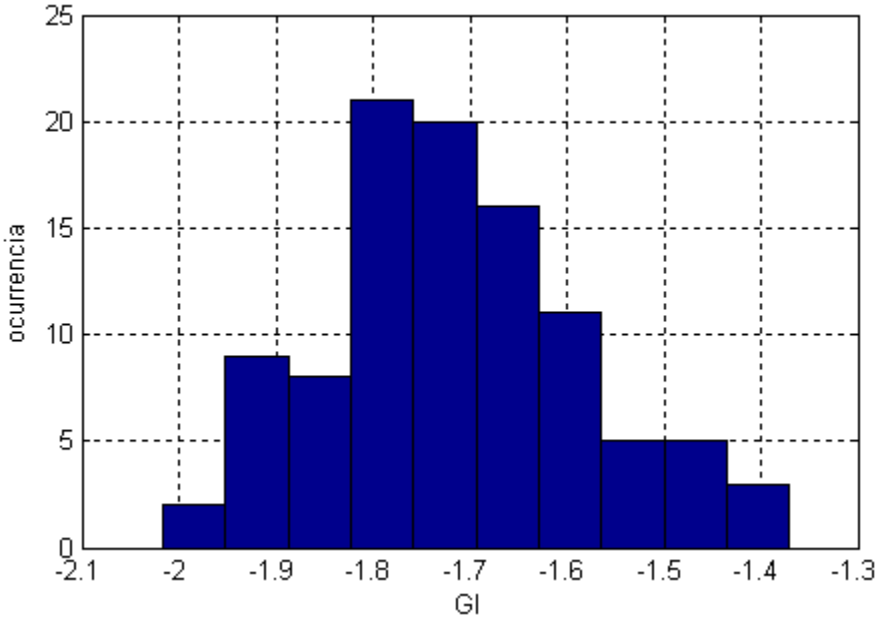


Figura 4.39 Distribución de *Baseline* para la imagen 14_2.tif de la base de datos 2 del FVC2002.

4.6. Resumen

En este capítulo nos hemos enfocado en detallar las técnicas de mayor relevancia para la determinación de aquellas características de una impresión dactilar que permiten tanto agruparlas (características globales) como reconocerlas (características locales), siendo éstas de utilidad para las etapas de clasificación y coincidencia respectivamente. También se ha expuesto una técnica alterna que permite extraer un tipo de característica híbrida que contiene simultáneamente información global y local, basada en los patrones de textura orientada que presenta la imagen denominada *Finger Code*.

En la sección dedicada a la determinación de los puntos singulares se han presentado dos técnicas, la primera basada en el índice de Poincaré (sección 4.2.1.1.) y la segunda basada en los componentes sinusoidales de la Imagen de Orientación (sección 4.2.1.2.). De los resultados experimentales se concluye que frente a la calidad y a la rotación de la imagen la segunda técnica demuestra ser más robusta, permitiendo además determinar de manera directa el punto de referencia de una impresión de tipo A, sin embargo la técnica basada en el índice de Poincaré pre-

senta un mejor comportamiento cuando las impresiones son de buena calidad Puesto que el tiempo de procesamiento y la complejidad computacional son mucho menores en la segunda técnica respecto a la primera, se recomienda su uso cuando se realice procesos de inscripción de usuarios en línea y su posible implementación en sistemas embebidos.

En la sección dedicada a la extracción de las minucias hemos propuesto una técnica para la determinación de sus atributos (tipo, posición y orientación) a partir de la imagen binaria vinculada a la impresión dactilar, ésta presenta las siguientes contribuciones: (a) el realzado del contraste ecualizando el histograma del primer plano de la imagen y su posterior mejora a través de la transformada de Fourier bidimensional antes de realizar la binarización adaptativa eliminando así, el paso de la estimación de la Imagen de Frecuencias recomendado en [Tico y Onnia, 2002]; (b) el filtrado de la imagen esqueletizada a través de una máscara de codificación para la eliminación de todos aquellos píxeles que puedan generar problemas al momento de detectar las minucias; (c) la elaboración de un algoritmo de seguimiento de crestas robusto basado en reglas heurísticas plasmadas en una tabla de búsqueda (ver Apéndice A) que permite la determinación de la orientación de una minucia y (d) un paso de post procesamiento de minucias basado en las relaciones que presentan las minucias falsas. La técnica propuesta ha sido validada a través del indicador de rendimiento denominado Índice de Virtud y tal como lo indica la tabla 4.2 es robusta y exacta.

En la sección dedicada a la determinación del *Finger Code* realizamos una descripción de cada uno de los pasos necesarios para su implementación y se especifican las condiciones a tomar en cuenta para la calibración de sus parámetros de mayor relevancia, excluimos resultados experimentales puesto que han sido reportados detalladamente en [Prabhakar, 2001] para la base de datos 9 del NIST y en [Aching y Rojas, 2003] para la base de datos 1 del FVC2002.

Capítulo V
Coincidencia de las Impresiones Dactilares

5.1. Introducción

Todo algoritmo de coincidencia de impresiones dactilares tiene como entradas dos impresiones a comparar (primaria y secundaria) y como salida una puntuación de similitud (un indicador entre 0 y 1) o una decisión binaria (coincide/no coincide). Sólo unos pocos algoritmos de coincidencia operan directamente sobre las imágenes en escala de grises; la mayoría de éstos hacen uso de un vector de características vinculado a cada una de las imágenes obtenidas en la etapa de extracción de características. Sin pérdida de generalidad denotamos las impresiones dactilares primarias y secundarias como $\mathbf{I}_P$ e $\mathbf{I}_S$ y sus respectivos vectores de características como $\mathbf{v}_P$ y $\mathbf{v}_S$. Usualmente los algoritmos para la extracción de características y coincidencia son bastante parecidos tanto para los problemas de verificación así como para identificación, debido a que el problema de identificación puede ser implementado como una ejecución secuencial de N coincidencias uno a uno entre pares de impresiones dactilares. El diseño de un algoritmo fiable de coincidencia es muy desafiante debido a que las impresiones son ligeramente diferentes cada vez que son capturadas (variaciones dentro de una clase). Los principales factores responsables para las variaciones dentro de una clase son resumidas a continuación:

Desplazamiento: Al ser la huella colocada sobre el sensor en diferentes adquisiciones se genera una traslación en la impresión dactilar resultante. Por ejemplo, una huella desplazada solamente 2 mm (imperceptible para el usuario) resulta en una traslación de alrededor de 40 píxeles en una imagen escaneada a una resolución de 500 dpi.

Rotación: Al ser la huella colocada y a su vez rotada involuntariamente con respecto a la superficie del sensor en diferentes adquisiciones, se observa en la práctica que dichas rotaciones son alrededor de ±20º con respecto a la orientación vertical.

Traslapamiento parcial: El desplazamiento y la rotación de la huella a menudo causan que parte del área de la huella dactilar quede fuera de la superficie de adquisición del sensor, resultando así en un pequeño traslapamiento entre las áreas del primer plano de $\mathbf{I}_P$ e $\mathbf{I}_S$. Este problema es particularmente serio para sensores que presenten una superficie de adquisición pequeña.

Distorsiones no lineales: Se presenta al mapear (adquirir) la superficie elástica convexa de la huella en un sensor plano. Este mapeo produce distorsión no lineal producto de la plasticidad de la piel. A menudo los algoritmos de coincidencia de impresiones dactilares no toman en cuenta las características de tal mapeo y consideran a la impresión dactilar como no distorsionada suponiendo que fue producida por una correcta colocación de la huella. La colocación de la huella es correcta cuando el usuario:

(a) Aproxima la huella hacia el sensor a través de un movimiento que es ortogonal a su superficie.

(b) No aplica tracción o torsión una vez que la huella toca la superficie del sensor.

Presión y condiciones de la piel: Los patrones de cresta de la huella son capturados con precisión a través de un contacto uniforme con la superficie del sensor, sin embargo la presión aplicada al momento de la adquisición, los cortes y la sequedad de la piel, el sudor, la suciedad, la grasa y la humedad del aire generan un contacto no uniforme.

Ruido: Es principalmente introducido por los sistemas de captura. Por ejemplo, los residuos dejados sobre la superficie del sensor, luego de la captura previa de la huella dactilar.

Errores en la extracción de características: Los algoritmos de extracción de características son imperfectos y a menudo introducen errores en las mediciones. Estos errores pueden aparecer en cualquier fase (por ejemplo, en la estimación de la Imagen de Orientación, de la Imagen de Regiones, del número, tipo y posición de las singularidades, entre otros). Algunos algoritmos para el mejoramiento, pueden introducir perturbaciones en la localización y dirección de las minucias verdaderas. Para imágenes de pobre calidad es común detectar un gran número de minucias espurias y no ser capaz de detectar las minucias verdaderas.

Un gran número de técnicas automáticas para determinar la coincidencia de impresiones dactilares han sido propuestas en la literatura del reconocimiento de patrones. La mayoría de éstas pueden clasificarse a grandes rasgos dentro de tres familias:

Técnicas basadas en la correlación: Dos impresiones dactilares son superpuestas y la correlación entre los correspondientes píxeles es calculada para diferentes alineamientos (variaciones en el desplazamiento y la rotación). Las técnicas basadas en correlación son descritas en la sección 5.2.

Técnicas basadas en las minucias: Es la más popular y usada, siendo la técnica base para la comparación de impresiones dactilares creada por los expertos. Las minucias son extraídas de dos diferentes impresiones y almacenadas como conjunto de puntos en un plano bidimensional. Esta técnica esencialmente consiste en la determinación del alineamiento entre el conjunto de minucias de la impresión primaria y la impresión secundaria, del cual resulta el número máximo de minucias coincidentes. Las sección 5.3 es dedicada a esta técnica.

Técnicas basadas en las características de las crestas: La extracción de minucias a partir de las impresiones dactilares de pobre calidad es una tarea compleja. Sin embargo, se pueden considerar otras características de los patrones de crestas, como por ejemplo la orientación local y la frecuencia, la forma de las crestas y la información de textura. Las técnicas que corresponden a esta familia comparan las impresiones dactilares en términos de características extraídas desde el

patrón de crestas. En principio, la coincidencia basada en la correlación y en las minucias deben ser consideradas como sub familias de las técnicas basadas en las características de las crestas, ya que la intensidad de los píxeles y las posiciones de las minucias son características propias de los patrones de cresta. En la sección 5.4 discutimos únicamente esta técnica de coincidencia que no usa ni minucias ni intensidades de píxeles si no una representación de una impresión dactilar alterna basada en la textura denominada *Finger Code*.

5.2. Técnicas basadas en correlación

Una manera de establecer la similitud entre $\mathbf{I}_S$ e $\mathbf{I}_P$ es a través de la suma de las diferencias cuadradas SSD entre las intensidades de sus correspondientes píxeles:

$$\begin{aligned} SSD(\mathbf{I}_S, \mathbf{I}_P) &= \|\mathbf{I}_S - \mathbf{I}_P\|^2 = (\mathbf{I}_S - \mathbf{I}_P).(\mathbf{I}_S - \mathbf{I}_P)^T \\ &= \|\mathbf{I}_S\|^2 + \|\mathbf{I}_P\|^2 - 2\mathbf{I}_S{}^T.\mathbf{I}_P \end{aligned} \tag{5.1}$$

Si los términos $\|\mathbf{I}_S\|$ y $\|\mathbf{I}_P\|$ son constantes, la diferencia entre las dos imágenes es minimizada cuando la correlación cruzada CC entre $\mathbf{I}_S$ y $\mathbf{I}_P$ es maximizada:

$$CC(\mathbf{I}_S, \mathbf{I}_P) = \mathbf{I}_S{}^T.\mathbf{I}_P \tag{5.2}$$

Note que la cantidad $-2.CC(\mathbf{I}_S, \mathbf{I}_P)$ aparece como el tercer término en la ecuación (5.1). La correlación cruzada (o simplemente correlación) es entonces una medida de la similitud entre dos imágenes. Debido a la traslación y la rotación que inevitablemente caracterizan a dos impresiones de una misma huella, su similitud no puede ser simplemente calculada por la superposición de $\mathbf{I}_S$ e $\mathbf{I}_P$ y la aplicación de la ecuación (5.2). Si $\mathbf{I}_P^{(\Delta i,\Delta j,\theta)}$ representan una rotación de $\mathbf{I}_P$ por un ángulo θ alrededor del origen (usualmente el centro de la imagen) y corrida $\Delta i, \Delta j$ píxeles en direcciones i y j respectivamente; entonces la similitud entre $\mathbf{I}_S$ e $\mathbf{I}_P$ puede ser medida como:

$$S(\mathbf{I}_S, \mathbf{I}_P) = \max_{\Delta i,\Delta j,\theta} \left\{ CC(\mathbf{I}_S, \mathbf{I}_P^{(\Delta i,\Delta j,\theta)}) \right\} \tag{5.3}$$

La aplicación directa de la ecuación (5.3) raramente conduce a resultados aceptables principalmente debido a los siguientes problemas:

1. Las distorsiones no lineales generan una diferencia significativa en términos de las estructuras globales en las impresiones de la misma huella.
2. Las condiciones de la piel generan diferencias en la brillantez, el contraste y crestas gruesas en impresiones de la misma huella.
3. Una aplicación directa de la ecuación (5.3) es computacionalmente muy costosa.

5.2.1. Coincidencia basada en la correlación local

Como se ha expuesto en el párrafo anterior tres son los problemas que presentan las técnicas basadas en correlación, una manera de afrontarlos es a través de la correlación local, ya que la distorsión es usualmente manejada por la selección de regiones locales a correlacionar en lugar de utilizar la imagen entera. El proceso se resume en los siguientes pasos:

1. **Selección de las regiones locales**: Se define las regiones locales de $\mathbf{I}_S$ (tamaño típico puede ser 24x24 o 32x32 píxeles), estando caracterizadas por su nivel de coherencia, el cual es un indicador de la presencia de singularidades en la impresión dactilar.
2. **Coincidencia de regiones locales:** Cada región local de $\mathbf{I}_S$ es correlacionada con la imagen entera $\mathbf{I}_P$. La correlación estimada en las diferentes regiones es combinada para obtener una medida de similitud entre ambas imágenes.

Veamos en detalle cada uno de estos pasos:

5.2.1.1. Selección de las regiones locales

Las regiones locales pueden ser definidas de varias maneras: (a) su unión cubre completamente a $\mathbf{I}_S$ y sus intersecciones son nulas (cobertura total sin traslapamiento); (b) su unión cubre completamente a $\mathbf{I}_S$ y están localmente traslapadas (cobertura total con traslapamiento); (c) únicamente ciertas regiones

de interés son seleccionadas desde $\mathbf{I}_S$. En [Bazen et al., 2000] se presentan tres diferentes criterios para identificar tales regiones selectivas en la imagen $\mathbf{I}_S$:

(a) Las regiones alrededor de las minucias.

(b) Las regiones donde las crestas presentan un bajo nivel de coherencia.

(c) Las regiones que exhiben una baja autocorrelación a otras localizaciones en la imagen $\mathbf{I}_S$.

Tomando en cuenta el trabajo realizado en [URL4] presentamos la técnica desarrollada por nosotros para la detección de las regiones locales basada en el nivel de irregularidad, el cual es descrito en los siguientes pasos:

1. Determinamos el Vector Gradiente para cada píxel (u,v) de la imagen $\mathbf{I}_S$, utilizando para ello las máscaras extendidas de Sobel o Prewitt [Pajares y De la Cruz, 2002, pp. 158-159].

2. Calculamos el nivel de coherencia $\mathrm{Coh}[\mathbf{I}_S(u,v)]$ a través de la relación (C.15); es de vital importancia dar valores adecuados al tamaño y la desviación estándar de la máscara Gaussiana $\mathbf{W}_{GS}$.

3. Valiéndonos de $\Re_S$ y $\mathrm{Coh}[\mathbf{I}_S(u,v)]$ determinamos el nivel de irregularidad.

$$\mathrm{Irr}[\mathbf{I}_S(u,v)] = (1 - \mathrm{Coh}[\mathbf{I}_S(u,v)]).\Re_S(u,v) \qquad (5.4)$$

4. Es obvio que las regiones alrededor de las singularidades locales presentan un alto nivel de irregularidad, para lo cual se realiza una exploración fila por fila sobre la matriz $\mathrm{Irr}[\mathbf{I}_S]$ para la detección de los picos, cada fila es filtrada con una máscara Gaussiana unidimensional para eliminar las pequeñas variaciones. Si el punto donde se detecta un pico presenta un nivel de irregularidad mayor que cierto umbral (0.4 en nuestro caso), entonces en dicho punto habrá una posible singularidad local (ver figura 5.1).

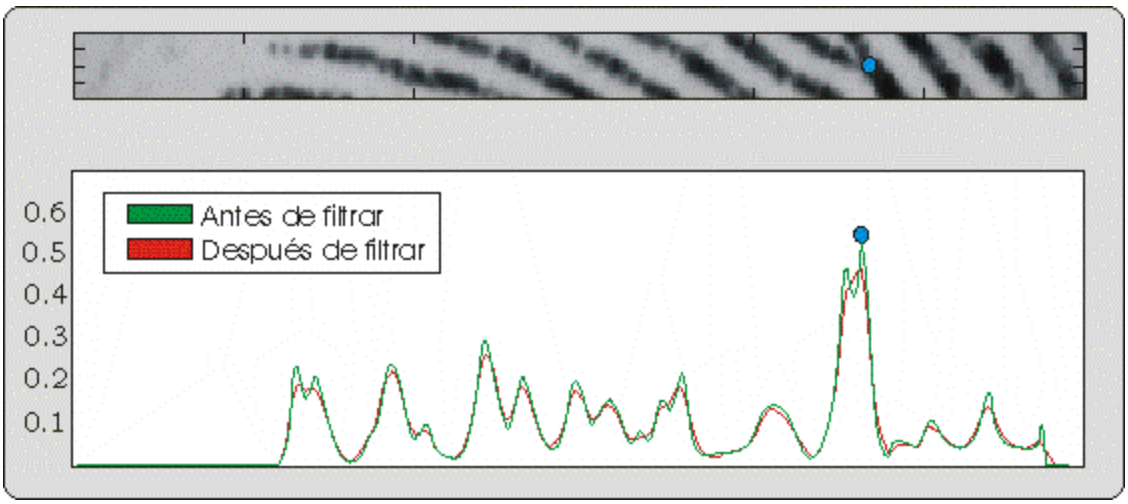


Figura 5.1 Picos presentes en una fila de la matriz de irregularidad y su vinculación con la posible posición de una singularidad local. Imagen 1_4.tif de la base de datos 2 del FVC2002.

5. El paso (4) se realiza hasta haber explorado todas las filas, obteniéndose al final un conjunto de puntos que definen las posibles posiciones de las singularidades locales, estos puntos son agrupados en N_R centroides (número de regiones locales que se desea) a través del algoritmo VQ. Los centroides generados por este algoritmo son asumidos como los centros geométricos de las regiones locales a extraer de $\mathbf{I}_S$, denotados por $\{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{N_R}\}$.

En la figura 5.2 se muestran las matrices $\mathrm{Coh}[\mathbf{I}_S]$ y $\mathrm{Irr}[\mathbf{I}_S]$, obtenidas en los pasos (2) y (3), utilizando una máscara Gaussiana de 15x15 píxeles con $\sigma = 2$ píxeles, además se muestran las regiones locales de 55x55 píxeles que toman como centro geométrico los centroides obtenidos al agrupar las singularidades locales.

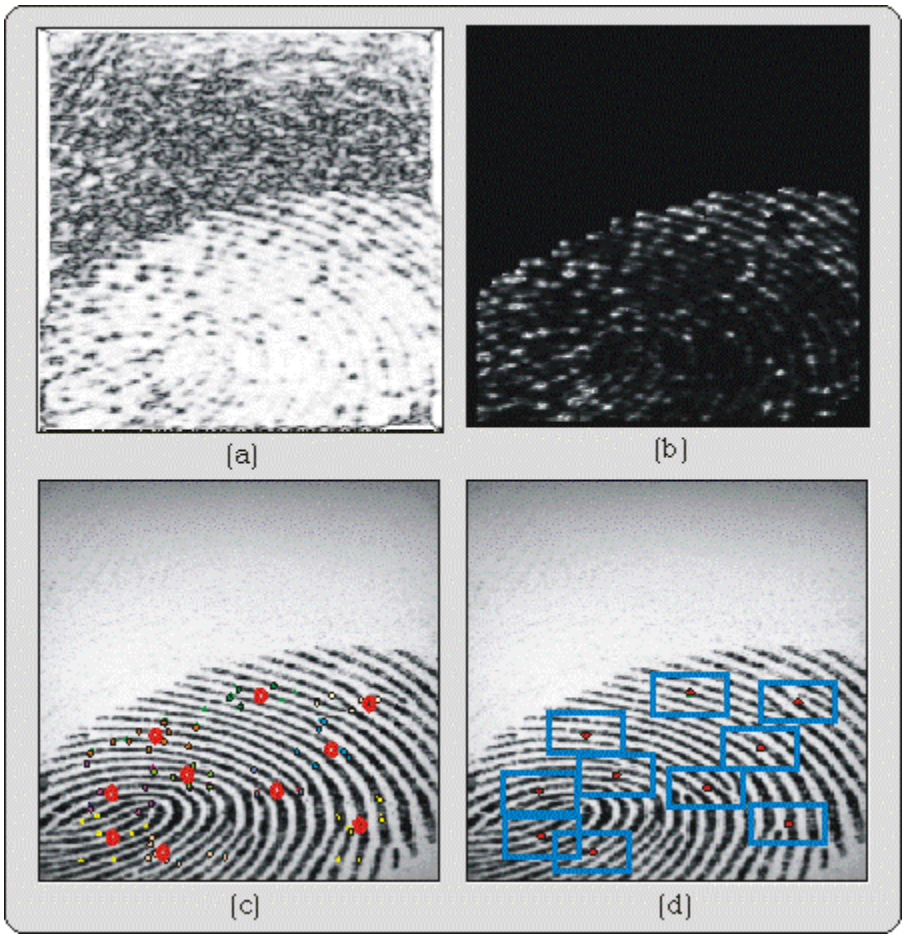


Figura 5.2 (a) Matriz de coherencia $\mathrm{Coh}[\mathbf{I}_S]$; (b) matriz de Irregularidad $\mathrm{Irr}[\mathbf{I}_S]$; (c) agrupamiento de las posibles singularidades locales en N_R (=10) centroides; (d) regiones locales $\{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{N_R}\}$. Imagen 1_4.tif de la base de datos 2 del FVC2002.

5.2.1.2. Coincidencia de regiones locales

Luego de obtener las regiones locales de $\mathbf{I}_S$ siguiendo el procedimiento descrito en la sección anterior, correlacionamos cada una de estas regiones con la imagen entera $\mathbf{I}_P$. Utilizamos para ello la correlación cruzada normalizada [Crouzil et al., 1996], ya que esta disminuye en gran medida el problema de las variaciones del contraste y brillo, además su adecuada implementación permite optimizar el tiempo de coincidencia. Este algoritmo ha sido utilizado para realizar reconocimiento de rostros, registrado de imágenes, entre otros [Rademacher, 2001] y tal como es indicado en [URL 5.1][URL 5.2] es una herramienta útil para la coinci-

dencia de patrones. Veamos a continuación una breve explicación de la correlación cruzada normalizada.

5.2.1.2.1. Correlación cruzada normalizada

Para el caso de la correlación local la suma de las distancias al cuadrado vendrá definida por la expresión:

$$\begin{aligned} SSD[\mathbf{I}(u,v),\mathbf{M}] &= \sum_{u'=u-\overline{N}}^{u+\overline{N}} \sum_{v'=v-\overline{M}}^{v+\overline{M}} \left(\mathbf{I}(u',v') - \mathbf{M}(u'-u,v'-v)\right)^2 \\ &= Mean[\mathbf{I}^2(u,v)] + Mean[\mathbf{M}^2] - 2.\sum_{u'=u-\overline{N}}^{u+\overline{N}} \sum_{v'=v-\overline{M}}^{v+\overline{M}} \mathbf{I}(u',v').\mathbf{M}(u'-u,v'-v) \end{aligned} \tag{5.5}$$

Donde:

$\mathbf{I}$: Es la imagen base de tamaño $N_I x M_I$.

$\mathbf{M}$: Es la máscara a correlacionar sobre $\mathbf{I}$ de tamaño NxM, tal que: $\overline{N} = N/2$ y $\overline{M} = M/2$.

$Mean[\mathbf{I}^2(u,v)]$: Es el valor promedio de $\mathbf{I}^2$ en la región centrada en (u,v).

$Mean[\mathbf{M}^2]$: Es el valor promedio de $\mathbf{M}^2$.

Si los términos $Mean[\mathbf{I}^2(u,v)]$ y $Mean[\mathbf{M}^2]$ son considerados constantes, la diferencia entre la imagen base $\mathbf{I}$ y la máscara $\mathbf{M}$ es minimizada cuando el tercer término en la ecuación (5.5) es maximizado, dicho término es denominado como la correlación cruzada estándar entre $\mathbf{I}$ y $\mathbf{M}$, siendo definida como:

$$CC[\mathbf{I}(u,v),\mathbf{M}] = \sum_{u'=u-\overline{N}}^{u+\overline{N}} \sum_{v'=v-\overline{M}}^{v+\overline{M}} \mathbf{I}(u',v').\mathbf{M}(u'-u,v'-v) \tag{5.6}$$

Usamos (5.6) para establecer la similitud entre la región extraída de la imagen $\mathbf{I}$ y la máscara $\mathbf{M}$ presenta varias desventajas, veamos:

1. Si $Mean[\mathbf{I}^2(u,v)]$ varía con la posición, la coincidencia usando (5.6) fallará.
2. El rango de $CC[\mathbf{I}(u,v),\mathbf{M}]$ es dependiente del tamaño de $\mathbf{M}$.
3. La ecuación (5.6) no es invariante a los cambios en la amplitud de la

imagen causadas por las variaciones de brillantez y contraste.

La correlación cruzada normalizada sobrelleva estas dificultades al normalizar la imagen $\mathbf{I}$ y la máscara $\mathbf{M}$, estando ésta definida como:

$$\mathrm{NCC}[\mathbf{I}(u,v),\mathbf{M}] = \frac{\sum_{u'=u-\bar{N}}^{u+\bar{N}}\sum_{v'=v-\bar{M}}^{v+\bar{M}}\left(\mathbf{I}(u',v')-\mathrm{Mean}[\mathbf{I}(u,v)]\right)\left(\mathbf{M}(u'-u,v'-v)-\mathrm{Mean}[\mathbf{M}]\right)}{\sqrt{\sum_{u'=u-\bar{N}}^{u+\bar{N}}\sum_{v'=v-\bar{M}}^{v+\bar{M}}\left(\mathbf{I}(u',v')-\mathrm{Mean}[\mathbf{I}(u,v)]\right)^2\sum_{u'=u-\bar{N}}^{u+\bar{N}}\sum_{v'=v-\bar{M}}^{v+\bar{M}}\left(\mathbf{M}(u'-u,v'-v)-\mathrm{Mean}[\mathbf{M}]\right)^2}} \qquad (5.7)$$

Donde:

$\mathrm{Mean}[\mathbf{I}(u,v)]$: Es el valor promedio de $\mathbf{I}$ en la región centrada en (u,v).

$\mathrm{Mean}[\mathbf{M}]$: Es el valor promedio de la máscara $\mathbf{M}$.

$\mathrm{NCC}[\mathbf{I}(u,v),\mathbf{M}]$: Es la correlación cruzada normalizada entre la región extraída de $\mathbf{I}$ centrada en (u,v) y la máscara $\mathbf{M}$. Se encuentra en el rango de $\leq 0,1 \geq$, valores cercanos a uno indican una alta similitud entre la región extraída de la imagen $\mathbf{I}$ y la máscara $\mathbf{M}$.

La correlación cruzada normalizada a diferencia de la correlación cruzada estándar es calculable únicamente en el dominio espacial, por lo que es computacionalmente costosa. Varias técnicas de cálculo inexactas pero rápidas han sido propuestas, siendo una de las más eficientes y exactas la propuesta por Lewis [URL 5.3], este hace uso de la siguiente representación simplificada de la correlación cruzada normalizada:

$$\mathrm{NCC}[\mathbf{I}(u,v),\mathbf{M}] = \frac{1}{\sqrt{\mathrm{N.M.(N.M-1)}}}\cdot\frac{\mathrm{CC}[\mathbf{I}(u,v),\mathbf{M}]-(\mathrm{N.M}).\mathrm{Mean}[\mathbf{M}].\mathrm{Mean}[\mathbf{I}(u,v)]}{\left|\mathrm{Std}[\mathbf{M}]\right|\sqrt{(\mathrm{Mean}[\mathbf{I}^2(u,v)]-\mathrm{Mean}^2[\mathbf{I}(u,v)])}} \qquad (5.8)$$

Podemos apreciar que son dos las operaciones a optimizar, la correlación cruzada $\mathrm{CC}[\mathbf{I}(u,v),\mathbf{M}]$ y las operaciones de promediación $\mathrm{Mean}[\mathbf{I}(u,v)]$ y $\mathrm{Mean}[\mathbf{I}^2(u,v)]$. El primer caso es resuelto a través del cálculo de la correlación cruzada estándar en el dominio de la frecuencia [Gonzáles y Woods, 1996, pp. 110-118], a través de la relación:

$$\begin{aligned} NCC[\mathbf{I}(u,v),\mathbf{M}] &= \mathbf{I}(u,v) * \mathbf{M}(-u,-v) \\ &= \text{IFFT2}\left\{\text{FFT2}\{\mathbf{I}(u,v)\}.\text{FFT2}\{\mathbf{M}(-u,-v)\}^{*}\right\} \end{aligned} \tag{5.9}$$

El segundo caso es resuelto elegantemente por Lewis al plantear la operación de promediación por vecindad sobre $\mathbf{I}$ como un proceso iterativo, veamos:

1. Determinamos la matriz de sumas acumuladas $\mathbf{SUM}$.

$$\begin{aligned} \mathbf{SUM}(u,v) &= \sum_{u'=0}^{u}\sum_{v'=0}^{v}\mathbf{I}(u',v') \\ &= \mathbf{I}(u,v) + \mathbf{SUM}(u-1,v) + \mathbf{SUM}(u,v-1) - \mathbf{SUM}(u-1,v-1) \end{aligned} \tag{5.10}$$

tal que:

$u = 0,1,\ldots,N_I - 1$ y $v = 0,1,\ldots,M_I - 1$ y

$\mathbf{SUM}(u,v) = 0$, para todo, $u < 0 \vee v < 0$.

2. Determinamos el valor promedio de $\mathbf{I}$ en la región centrada en (u,v) a través de la relación:

$$\begin{aligned} \text{Mean}[\mathbf{I}(u,v)] &= \sum_{u'=u-\overline{N}}^{u+\overline{N}}\sum_{v'=v-\overline{M}}^{v+\overline{M}}\mathbf{I}(u',v') \\ &= \mathbf{SUM}(u+\overline{N},v+\overline{M}) - \mathbf{SUM}(u-\overline{N},v-\overline{M}-1) \\ &\quad - \mathbf{SUM}(u-\overline{N}-1,v+\overline{M}) + \mathbf{SUM}(u-\overline{N}-1,v-\overline{M}-1) \end{aligned} \tag{5.11}$$

El número total de operaciones que se realiza en los dos pasos anteriores es de $6.N^I.M^I$, el cual representa un menor costo computacional frente a la técnica común de promediación que implica realizar $(N-1)(M-1).N^I.M^I$ operaciones.

5.2.1.2.2. Coincidencia de regiones a través de la correlación cruzada normalizada

La correlación cruzada normalizada estimada en las regiones $\{\mathbf{M}_1, \mathbf{M}_2, \ldots, \mathbf{M}_{N_R}\}$ produce un conjunto de pares coordenados denotados por:

$$POS_P = \{(u_1^p, v_1^p); (u_2^p, v_2^p); \ldots; (u_{N_R}^p, v_{N_R}^p)\} \quad (5.12)$$

$$POS_S = \{(u_1^s, v_1^s); (u_2^s, v_2^s); \ldots; (u_{N_R}^s, v_{N_R}^s)\} \quad (5.13)$$

Donde: (u_k^p, v_k^p) es la posición en la imagen $\mathbf{I}_P$ cuya vecindad produce la máxima correlación con la región $\mathbf{M}_k$ centrada en la posición (u_k^s, v_k^s). Ahora para establecer el nivel de similitud entre los N_R pares hacemos uso de la correspondencia entre posiciones relativas de pares PRP, definidas como :

$$\left[(u_i^s, v_i^s) - (u_j^s, v_j^s)\right] \approx \left[(u_i^p, v_i^p) - (u_j^p, v_j^p)\right] \quad (5.14)$$

Donde: $1 \le i, j \le N_R$ y $i \ne j$

Las PRP son convertidas a coordenadas polares a través de las relaciones:

$$\rho_{ij} = \sqrt{(u_i - u_j)^2 + (v_i - v_j)^2} \quad (5.15)$$

$$\varphi_{ij} = \text{arctg2}(u_i - u_j, v_i - v_j) \quad (5.16)$$

donde $1 \le i, j \le N_R$ y $i \ne j$

Esto nos conduce a la prueba de correspondencia:

$$(\rho_{ij}^s, \varphi_{ij}^s) \stackrel{?}{\approx} (\rho_{ij}^p, \varphi_{ij}^p) \text{; para todo: } 1 \le i, j \le N_R \text{ e } i \ne j \quad (5.17)$$

El cual tolera cierta rotación, escalamiento y distorsiones no uniformes entre pares, causadas por la elasticidad de la huella dactilar. El siguiente paso es la clasificación de los PRP como coincidentes o no coincidentes a través de la aplicación de un umbral elíptico, es decir:

$$\left(\frac{\ln\left|\rho_{ij}^{s}/\rho_{ij}^{p}\right|}{\rho_T}\right)^2+\left(\frac{\varphi_{ij}^{s}-\varphi_{ij}^{p}}{\varphi_T}\right)^2<1 \qquad (5.18)$$

Donde, los umbrales ρ_T y φ_T definen el grado de tolerancia que pueden ser establecidos. En la figura 5.3 podemos ver un ejemplo de la relación entre los puntos coincidentes obtenidos en la etapa de correlación y la gráfica de dispersión de sus PRP asociados, puede apreciarse que una gran cantidad de éstos se encuentran dentro de la elipse con umbrales de $\rho_T=0.5$ y $\varphi_T=\pi/5$

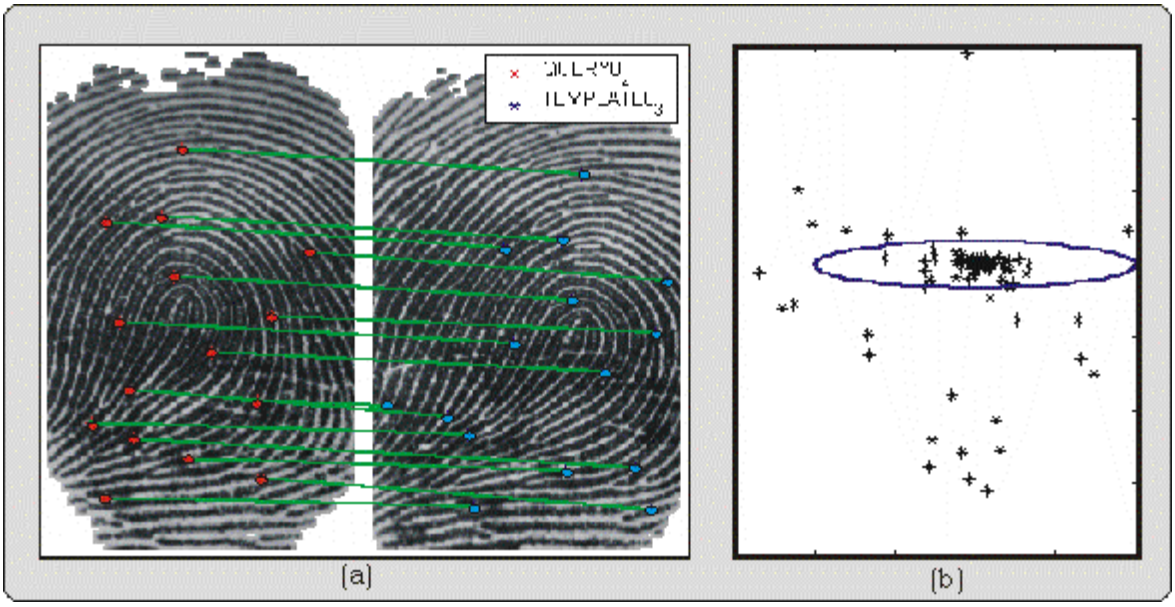


Figura 5.3 (a) Coincidencia entre puntos correlacionados POS_P y POS_S; (b) gráfica de dispersión $\ln\left|\rho_{ij}^{s}/\rho_{ij}^{p}\right|$ versus $\varphi_{ij}^{s}-\varphi_{ij}^{p}$. Imagen 8_2.tif y 8_4.tif de la base de datos 2 del FVC2002.

5.3. Técnica basada en las minucias

La coincidencia de minucias es ciertamente la técnica más conocida y utilizada, gracias a su estricta analogía con la forma en que los expertos comparan manualmente las impresiones dactilares.

Como se comentó en la sección 4.3.1.3.2. cada minucia es descrita por una tripleta de atributos, compuesta por su ubicación y dirección, siendo éstas agru-

padas por el tipo al que pertenezcan, algunas representaciones incluyen un peso que indica la calidad de la imagen en la vecindad de la minucia.

Agrupando todas las minucias tanto para la imagen $\mathbf{I}_P$ e $\mathbf{I}_S$, obtendremos sus respectivos vectores de características denotados por:

$$\begin{aligned}\upsilon_P &= \{MF_P, MB_P\} \\ &= \{m_1, m_2, \ldots, m_{N_{MP}}\},\ m_i = \{x_i, y_i, \theta_i\},\ i = 1, \ldots, N_{MP}\end{aligned} \tag{5.19}$$

$$\begin{aligned}\upsilon_S &= \{MF_S, MB_S\} \\ &= \{m'_1, m'_2, \ldots, m'_{N_{MS}}\},\ m'_j = \{x'_j, y'_j, \theta'_j\},\ j = 1, \ldots, N_{MS}\end{aligned} \tag{5.20}$$

Donde N_{MP} y N_{MS} denotan el número de minucias en $\mathbf{I}_P$ e $\mathbf{I}_S$ respectivamente. Una minucia m_i en $\mathbf{I}_P$ y una minucia m'_j en $\mathbf{I}_S$ son consideradas "coincidentes", si la distancia espacial entre éstas es menor a una tolerancia radial r_0 y la diferencia angular es más pequeña que una tolerancia angular θ_0, es decir:

$$sd(m'_j, m_i) = \sqrt{(x'_j - x_i)^2 + (y'_j - y_i)^2} \le r_{0.} \tag{5.21}$$

$$dd(m'_j, m_i) = \min(|\theta'_j - \theta_i|, 360° - |\theta'_j - \theta_i|) \le \theta_{0.} \tag{5.22}$$

La ecuación 5.22 toma el mínimo valor de $|\theta'_j - \theta_i|$ y $360° - |\theta'_j - \theta_i|$, por la periodicidad de los ángulos (la diferencia entre los ángulos de 2° y 358° es sólo 4°). Los círculos de tolerancia (o hiperesferas) definidos por r_0 o θ_0 son necesarios para compensar los errores inevitables creados por los algoritmos de extracción de características y por las distorsiones plásticas que causan la variación en la posición de las minucias.

Alinear las dos impresiones dactilares es un paso obligatorio para maximizar el número de minucias coincidentes, llevarlo a cabo correctamente requiere la aplicación de transformaciones geométricas a los conjuntos de puntos, como son las transformaciones de rotación, traslación y escalamiento (el escalamiento se considera cuando la resolución de ambas impresiones son diferentes, por ejemplo,

en el caso de que las dos huellas dactilares hayan sido tomadas por dos escáneres que operan a diferentes resoluciones).

Sea $\mathrm{map}(\bullet)$ la función que mapea la minucia m'_j (de $\mathbf{I}_S$) como m''_j de acuerdo con la transformación de similitud dada; por ejemplo, considerando un desplazamiento de $(\Delta x, \Delta y)$ y una rotación antihoraria θ alrededor del origen, tendremos que:

$$\mathrm{map}_{(\Delta x,\Delta y,\theta)}\left(m'_j = \{x'_i, y'_i, \theta'_i\}\right) = m''_j = \{x''_j, y''_j, \theta''_j + \theta\} \tag{5.23}$$

Donde:

$$\begin{bmatrix} x''_j \\ y''_j \end{bmatrix} = \begin{bmatrix} -\cos\theta & \mathrm{sen}\theta \\ \mathrm{sen}\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x'_j \\ y'_j \end{bmatrix} + \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \tag{5.24}$$

Sea $\mathrm{mm}(\bullet)$ la función que retorna el valor uno en el caso donde las minucias m''_j y m_i coincidan de acuerdo a las ecuaciones (5.21) y (5.22):

$$\mathrm{mm}\left(m''_j, m_i\right) = \begin{cases} 1 & \mathrm{sd}\left(m''_j, m_i\right) \le r_0 \text{ y } \mathrm{dd}\left(m''_j, m_i\right) \le \theta_0 \\ 0 & \text{en caso contrario} \end{cases} \tag{5.25}$$

Entonces, el problema de coincidencia puede ser formulado como sigue:

$$\max_{\Delta x,\Delta y,\theta P} \left\{ \sum_{i=1}^{m} \mathrm{mm}\left(\mathrm{map}_{\Delta x,\Delta y,\theta}\left(m'_{\mathrm{prd}(i)}\right), m_i\right) \right\} \tag{5.26}$$

Donde $\mathrm{prd}(i)$ es una función desconocida que determina el apareamiento entre las minucias de $\mathbf{I}_P$ e $\mathbf{I}_S$; en particular cada minucia tendrá una minucia correspondiente, o no tendrá correspondencia alguna, tal que si:

1. $\mathrm{prd}(i) = j$ indica que la minucia m_i en $\mathbf{I}_P$ coincide con la minucia m'_j en $\mathbf{I}_S$.
2. $\mathrm{prd}(i) = 0$ indica que la minucia m_i en $\mathbf{I}_P$ no coincide con ninguna minucia de $\mathbf{I}_S$.
3. $\mathrm{prd}(i) \neq j$ indica que la minucia m_i en $\mathbf{I}_P$ no coincide con la minucia m'_j

en $\mathbf{I}_S$.

Note que en general $\text{prd}(i) = j$ no necesariamente significa que las minucias m'_j y m_i coincidan en el sentido de las ecuaciones (5.21) y (5.22) pero sólo ellas conforman el par más probable bajo la transformación actual.

La expresión (5.26) requiere que el número de minucias apareadas sea maximizado, independientemente de que tan estricto sea llevado acabo el apareamiento.

La solución del problema de la coincidencia de minucias (5.26) es trivial cuando se conoce el correcto alineamiento $(\Delta x, \Delta y, \theta)$, de echo el apareamiento (por ejemplo, la función $\text{prd}(\bullet)$) puede ser determinado si establecemos para cada $i = 1,\ldots,N_{MP}$ las siguientes condiciones:

$\text{prd}(i) = j$ si $m''_j = \text{map}_{\Delta x,\Delta y,\theta}(m'_j)$ es el más cercano a m_i entre las minucias $\left\{m''_k = \text{map}_{\Delta x,\Delta y,\theta}(m'_k) \middle| k = 1,\ldots,N_{MS}, \text{mm}(m''_k, m_i) = 1\right\}$.

$\text{prd}(i) = 0$ si, para todo, $k = 1,\ldots,N_{MS}$, se cumple que:
$\text{mm}\left(\text{map}_{\Delta x,\Delta y,\theta}(m'_k), m_i\right) = 0$

Para cumplir con las dos restricciones vistas en el párrafo anterior, cada minucia m''_j ya apareada debe ser etiquetada para evitar aparearla una vez más, En la figura 5.4 se muestra un ejemplo de un apareamiento de minucias después de haber realizado el alineamiento de puntos. Las minucias de $\mathbf{I}_S$ están referidas como $m''_{\bullet}$ debido a que están trazadas en las coordenadas de $\mathbf{I}_P$. El apareamiento es realizado de acuerdo a la distancia mínima. Los círculos punteados indican la máxima distancia espacial. Los círculos grises denotan un apareamiento satisfactorio; la minucia m_1 de $\mathbf{I}_P$ y la minucia m''_3 de $\mathbf{I}_S$ no tienen pares coincidentes, la minucia de m_3 y m''_6 no forman un par coincidente debido a sus grandes diferencias de dirección.

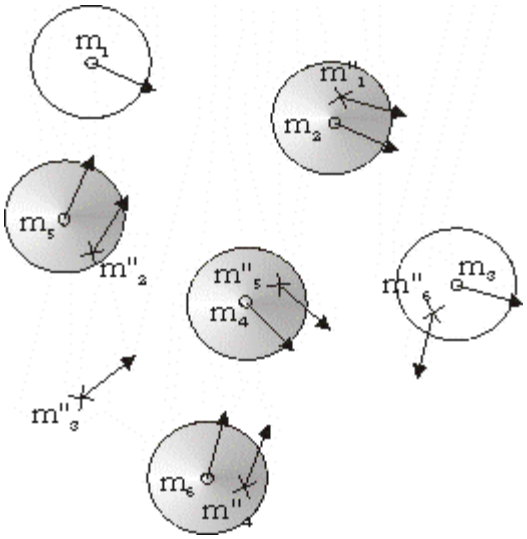


Figura 5.4 Apareamiento de minucias. Las minucias de $\mathbf{I}_P$ son denotadas por "o", mientras que las minucias $\mathbf{I}_S$ son denotadas por "x".

Para alcanzar un apareamiento óptimo (de acuerdo con la ecuación (5.26)) se adopta un esquema ligeramente más complicado, de echo en el caso donde una minucia de $\mathbf{I}_S$ cae dentro de la hiperesfera de tolerancia de más de una minucia de $\mathbf{I}_P$, la asignación correcta es la que maximiza el número de pares coincidentes, en el ejemplo de la figura 5.5 si m_1 fuera apareada con m''_2 (la minucia más cercana) m_2 se quedaría sin par coincidente, sin embargo apareando m_1 con m''_1, se permite que m_2 se aparee con m''_2, de esta manera, apareamos según la ecuación (5.26).

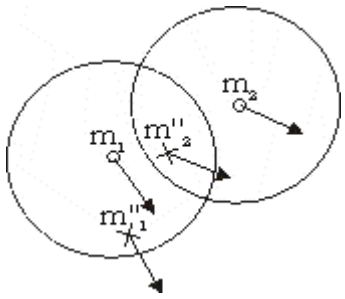


Figura 5.5 Problemas para un apareamiento óptimo.

La maximización de (5.26) puede ser fácilmente resuelta si la función $\mathrm{prd}(\bullet)$ (correspondencia de minucias) es conocida. En este caso el alineamiento desconocido $(\Delta x, \Delta y, \theta)$ puede ser determinado a través de mínimos cuadrados [Umeyana, 1991] y [Chang et al., 1997]. Desafortunadamente, en la práctica ninguno de los

parámetros de alineamiento ni la función de correspondencia $\mathrm{prd}(\bullet)$, son conocidos y además resolver el problema de coincidencia es muy difícil. Una técnica de "Fuerza bruta", que evalúa todas las posibles soluciones (correspondencias y alineamientos) es prohibitiva puesto que el número de posibles soluciones es exponencial al número de minucias. Pocas técnicas de "Fuerza Bruta" han sido propuestas, por ejemplo [Huvanandana et al., 2000] propuso la cuantización ordinaria para las localizaciones de las minucias y la realización de una búsqueda exhaustiva para determinar el alineamiento óptimo.

En la literatura del reconocimiento de patrones, el problema de coincidencia de minucias ha sido generalmente tratado como un problema de coincidencia de patrones de puntos. Los dos problemas pueden ser aproximados a pesar de la pequeña diferencia que existe por la presencia de la dirección asociada con cada minucia. Debido a su rol central en muchas tareas para el reconocimiento de patrones y visión computacional (por ejemplo: coincidencia de objetos, control remoto, calibración de cámaras, estimación de movimiento, entre otros), la coincidencia de patrones de puntos ha sido estudiada en detalle generando una familia de técnicas conocidas como técnicas de relajación, algebraicas, de búsqueda de soluciones operacionales, de poda de árbol, de minimización de energía, basadas en la transformada de Hough, entre muchas otras. En este trabajo hemos desarrollado una técnica de coincidencia de patrones de puntos basada en la triangulación de Delaunay que permite definir una estructura topológica (triangulación de puntos) vinculada a las posiciones de las minucias a través de la cual se establece el alineamiento de los conjuntos de minucias a comparar, paso necesario para la coincidencia de minucias.

5.3.1. Técnica de coincidencia basada en el alineamiento de minucias a través de la triangulación de Delaunay

La triangulación es un proceso que toma una región de un espacio y la divide en un conjunto de sub regiones, el espacio puede ser de una o más dimensiones, sin embargo consideramos un espacio bidimensional puesto que tratamos con puntos (las posiciones de las minucias). En este caso las sub regiones son

simplemente triángulos. La triangulación tiene muchas aplicaciones en simulación de elementos finitos, aproximación de superficies, identificación de vecinos cercanos, entre otros. El objetivo del uso de la triangulación de Delaunay en este trabajo es la asociación de una estructura topológica bidimensional vinculada a las posiciones de las minucias.

Dado un conjunto de puntos $S_V = \{v_1, v_2, \cdots, v_N\}$ podemos obtener su triangulación de Delaunay al calcular primero el correspondiente diagrama de Voronoi, que descompone el espacio bidimensional en sub regiones centradas alrededor de cada punto v_i tal que todos los puntos en la región alrededor de v_i son los más cercanos a v_i que a cualquier otro punto en S_V. Dado el diagrama de Voronoi, la triangulación de Delaunay puede ser obtenida por la conexión de los centros de todos los pares de las regiones de Voronoi. La figura 5.6.a muestra un conjunto de puntos bidimensional. Su diagrama de Voronoi es mostrado en la figura 5.6.b, mientras que su triangulación de Delaunay es mostrado en la figura 5.6.c.

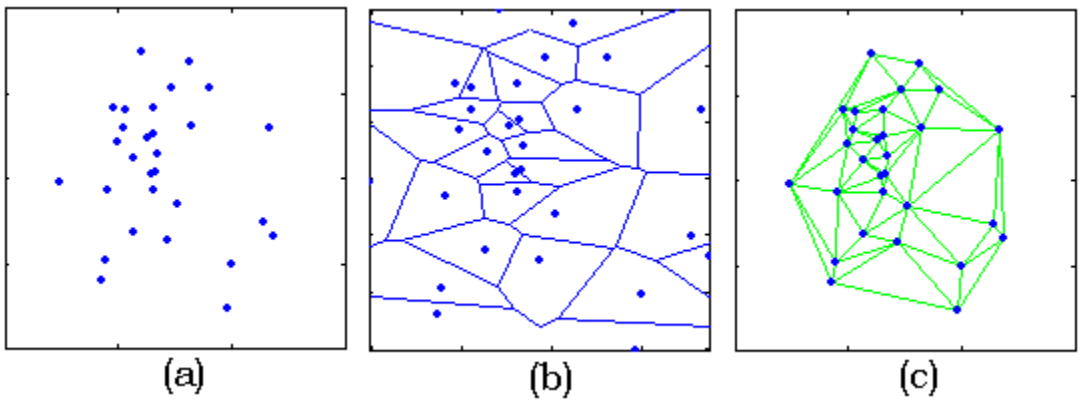


Figura 5.6 (a) conjunto de puntos bidimensional; (b) diagrama de Voronoi; (c) triangulación de Delaunay.

La triangulación de Delaunay tendrá ciertas propiedades, que comprenden:

1. La triangulación de un conjunto de puntos no degenerados es única.
2. Un círculo alrededor de tres puntos de todo triángulo de Delaunay no contiene otros puntos.

3. El ángulo mínimo frente a todos los ángulos en todos los triángulos en una triangulación de Delaunay es mayor que el ángulo mínimo en cualquier otra triangulación de los mismos puntos.

La primera propiedad sustenta el uso de la triangulación de Delaunay como una técnica idónea para la obtención de una estructura topológica bidimensional única que represente a los conjuntos de minucias. La segunda propiedad implica que la inserción de nuevos puntos en un triángulo de Delaunay afecta únicamente al triángulo cuyos círculos contienen al punto. Como resultado, los efectos del ruido en la triangulación de Delaunay se dan únicamente en forma local. Esto es muy importante en el contexto de nuestra aplicación. La última propiedad implica que los triángulos obtenidos no son "delgados". Esto es también muy deseable en nuestra aplicación puesto que el cálculo de las transformaciones geométricas entre impresiones dactilares está basado en los triángulos de minucias. Usar triángulos "delgados" puede llevar a inestabilidades y errores.

El diagrama de Voronoi y la triangulación de Delaunay son algoritmos muy eficientes puesto que el número de bordes para ambos es proporcional al número de puntos y el número de triángulos generados por la triangulación de Delaunay es lineal al número de puntos (cada borde pertenece a lo sumo a dos triángulos o polígonos).

5.3.1.1. Coincidencia basada en la triangulación de Delaunay

Dada una impresión primaria $\mathbf{I}_P$ y una impresión candidata o secundaria $\mathbf{I}_S$, el paso de coincidencia tiene por finalidad determinar un índice de similitud entre $\mathbf{I}_P$ e $\mathbf{I}_S$, es decir, una medida que indique la semejanza de cada imagen $\mathbf{I}_P$ a la imagen secundaria $\mathbf{I}_S$. Como es obvio tal comparación es realizada en base a las minucias extraídas de $\mathbf{I}_P$ e $\mathbf{I}_S$. Para la determinación de dicho índice de similitud bajo la técnica propuesta es necesario ejecutar los siguientes pasos:

1. **Determinación de los triángulos de Delaunay**. Una vez que las minucias de $\mathbf{I}_P$ e $\mathbf{I}_S$ son extraídas (formándose los vectores característicos $\mathbf{\upsilon}_P$ y $\mathbf{\upsilon}_S$, respectivamente), se calcula la triangulación de Delaunay para cada conjunto

de minucias, generándose dos conjuntos de tripletas de puntos ζ_P y ζ_S que hacen referencia a los triángulos de Delaunay vinculados a υ_P y υ_S.

2. **Determinación de las Invarianzas de Triangulación**. Se obtienen las características geométricas invariantes a la rotación, traslación y escalamiento vinculadas a cada triángulo referenciado por ζ_P y ζ_S. generándose los conjuntos ξ_P y ξ_S denominados Invarianzas de Triangulación.

3. **Coincidencia de las Invarianzas de Triangulación**. Utilizando las imágenes $\mathbf{I}_P$ e $\mathbf{I}_S$ así como los conjuntos ξ_P y ξ_S se determinan los triángulos de Delaunay de ζ_P y ζ_S más parecidos, calculándose la transformación de similitud correspondiente a cada par de triángulos coincidentes.

4. **Determinación del índice de coincidencia**. Cada transformación de similitud obtenida es aplicada al conjunto de minucias υ_S, alineándose respecto al conjunto de minucias υ_P, luego se contabiliza el número de minucias apareadas, determinándose así el índice de similitud vinculada a cada transformación. El índice de similitud final es el máximo de todos éstos.

Veamos en detalle cada uno de estos pasos:

5.3.1.1.1. Determinación de los triángulos de Delaunay

Sin el uso de la triangulación de Delaunay, tendríamos que considerar todos los posibles triángulos. Asumiendo N minucias como el promedio, el número posible de triángulos es $O(N^3)$. En contraste, la triangulación de Delaunay genera únicamente O(N) triángulos. En nuestros experimentos, hemos usado el algoritmo Quickhull (ver [URL 5.4]) para la obtención de la triangulación de Delaunay, Los pasos necesarios para determinar la triangulación de Delaunay de las minucias de la imagen primaria y secundaria son:

1. Se detectan las minucias de $\mathbf{I}_P$ e $\mathbf{I}_S$ generándose los vectores característicos υ_P y υ_S.

2. Tomando las posiciones de las minucias de los vectores característicos υ_P y υ_S como puntos para la triangulación de Delaunay, obtenemos dos

conjuntos de tripletas de puntos ζ_P y ζ_S que hacen referencia a los triángulos de Delaunay correspondientes a cada conjunto de minucias.

$$\zeta_P = \left\{tr_1, tr_2, \ldots, tr_{N_{TRP}}\right\},\ tr_i = \left\{(x_{i_1}, y_{i_1}), (x_{i_2}, y_{i_2}), (x_{i_3}, y_{i_3})\right\} \quad (5.27)$$

$$\zeta_S = \left\{tr'_1, tr'_2, \ldots, tr'_{N_{TRS}}\right\},\ tr'_j = \left\{(x'_{j_1}, y'_{j_1}), (x'_{j_2}, y'_{j_2}), (x'_{j_3}, y'_{j_3})\right\} \quad (5.28)$$

En la figura 5.7 se muestra la triangulación de Delaunay vinculada a las minucias extraídas de dos imágenes pertenecientes a la base de datos 2 del FVC 2002.

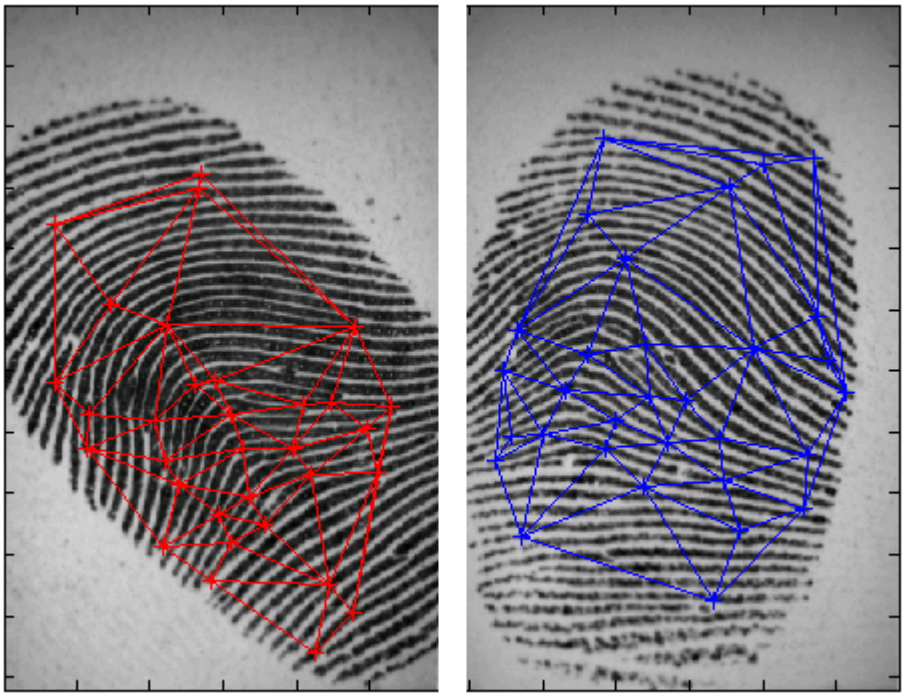


Figura 5.7 Triangulación de Delaunay a partir de las posiciones de las minucias. Imágenes 10_3.tif y 10_5.tif de la base de datos 2 FVC2002.

5.3.1.1.2. Determinación de las Invarianzas de Triangulación

Dado un triángulo de Delaunay (ver la figura 5.8), podremos calcular tres características invariantes a la rotación y traslación (del triángulo en si) las cuales son usadas para establecer la similitud entre triángulos. Las invarianzas están basadas en la longitud que tengan los lados. Los pasos para el cálculo de las Invarianzas de Triangulación son:

1. Para cada triángulo referenciado por una tripleta de puntos, se determina la longitud de sus lados y se calcula el ángulo de mayor abertura.
2. Si el ángulo calculado es mayor que un valor umbral (168º en nuestro caso) el triángulo es eliminado de las tripletas de puntos, en caso contrario los lados son ordenados en forma ascendente y los puntos de la tripleta son reordenados de manera tal que el punto vinculado al ángulo con mayor abertura ocupe la posición intermedia de la tripleta.

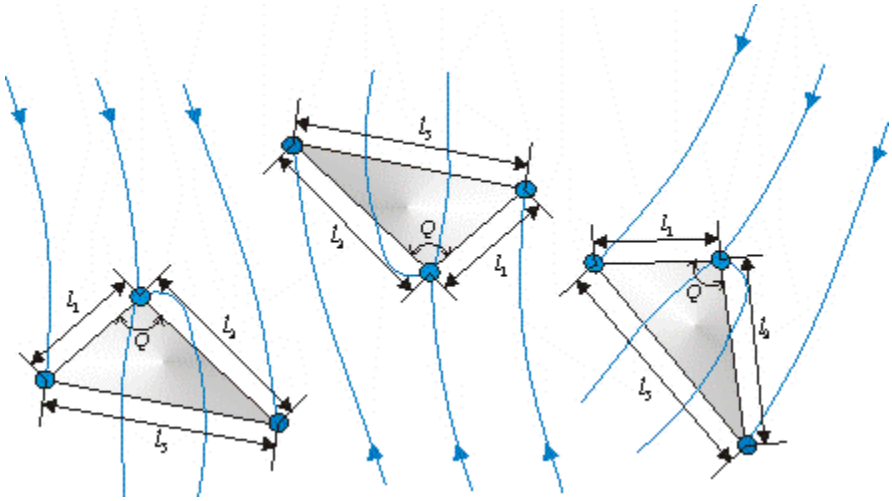


Figura 5.8 Invarianzas usando las minucias de triangulación.

Rechazamos los triángulos cuyos ángulos centrales son muy grandes puesto que el cálculo de las transformaciones geométricas en base a estos triángulos se vuelven inestables (pequeños errores en las posiciones de las minucias generan grandes errores en el cálculo de los parámetros de la transformación). Aunque la triangulación de Delaunay tiende a evitar los triángulos "delgados", esto no siempre es garantizado (a menos que puntos extras sean insertados en el conjunto de minucias).

Luego de haber realizado este paso se habrán obtenido dos conjuntos de tripletas vinculadas a las imágenes $\mathbf{I}_P$ e $\mathbf{I}_S$ que representan las Invarianzas de Triangulación denotadas por ξ_P y ξ_S, generándose además nuevas tripletas de puntos denotadas por $\overline{\zeta}_P$ y $\overline{\zeta}_S$. Es decir:

$$\begin{aligned}&\xi_{\mathrm{P}}=\left\{\mathrm{inv}_1,\mathrm{inv}_2,\ldots,\mathrm{inv}_{\mathrm{N_{INVP}}}\right\},\\&\mathrm{inv_i}=\left\{l_{\mathrm{i_1}},l_{\mathrm{i_2}},l_{\mathrm{i_3}}\right\},\ l_{\mathrm{i_1}}\le l_{\mathrm{i_2}}\le l_{\mathrm{i_3}}\end{aligned}\tag{5.29}$$

$$\begin{aligned}&\xi_{\mathrm{S}}==\left\{\mathrm{inv'}_1,\mathrm{inv'}_2,\ldots,\mathrm{inv'}_{\mathrm{N_{INVS}}}\right\},\\&\mathrm{inv_j}=\left\{l'_{\mathrm{i_1}},l'_{\mathrm{i_2}},l'_{\mathrm{i_3}}\right\},\ l'_{\mathrm{i_1}}\le l'_{\mathrm{i_2}}\le l'_{\mathrm{i_3}}\end{aligned}\tag{5.30}$$

$$\begin{aligned}&\overline{\zeta}_{\mathrm{P}}=\left\{\overline{\mathrm{tr}}_1,\overline{\mathrm{tr}}_2,\ldots,\overline{\mathrm{tr}}_{\mathrm{N_{INVP}}}\right\},\\&\overline{\mathrm{tr_i}}=\left\{(\mathrm{x}_{\mathrm{i_1}},\mathrm{y}_{\mathrm{i_1}}),(\mathrm{x}_{\mathrm{i_2}},\mathrm{y}_{\mathrm{i_2}}),(\mathrm{x}_{\mathrm{i_3}},\mathrm{y}_{\mathrm{i_3}})\right\},\ 0\le Q_i\le 168^{\mathrm{o}}\end{aligned}\tag{5.31}$$

$$\begin{aligned}&\overline{\zeta}_{\mathrm{S}}=\left\{\overline{\mathrm{tr}}'_1,\overline{\mathrm{tr}}'_2,\ldots,\overline{\mathrm{tr}}'_{\mathrm{N_{INVS}}}\right\},\\&\overline{\mathrm{tr}}'_{\mathrm{j}}=\left\{(\mathrm{x}'_{\mathrm{j_1}},\mathrm{y}'_{\mathrm{j_1}}),(\mathrm{x}'_{\mathrm{j_2}},\mathrm{y}'_{\mathrm{j_2}}),(\mathrm{x}'_{\mathrm{j_3}},\mathrm{y}'_{\mathrm{j_3}})\right\},\ 0\le Q'_j\le 168^{\mathrm{o}}\end{aligned}\tag{5.32}$$

Donde:

$$l_{\mathrm{i_1}}=\sqrt{\left(\mathrm{x}_{\mathrm{i_1}}-\mathrm{x}_{\mathrm{i_2}}\right)^2+\left(\mathrm{y}_{\mathrm{i_1}}-\mathrm{y}_{\mathrm{i_2}}\right)^2},\ l'_{\mathrm{i_1}}=\sqrt{\left(\mathrm{x}'_{\mathrm{i_1}}-\mathrm{x}'_{\mathrm{i_2}}\right)^2+\left(\mathrm{y}'_{\mathrm{i_1}}-\mathrm{y}'_{\mathrm{i_2}}\right)^2}\tag{5.33}$$

$$l_{\mathrm{i_2}}=\sqrt{\left(\mathrm{x}_{\mathrm{i_3}}-\mathrm{x}_{\mathrm{i_2}}\right)^2+\left(\mathrm{y}_{\mathrm{i_3}}-\mathrm{y}_{\mathrm{i_2}}\right)^2},\ l'_{\mathrm{i_2}}=\sqrt{\left(\mathrm{x}'_{\mathrm{i_3}}-\mathrm{x}'_{\mathrm{i_2}}\right)^2+\left(\mathrm{y}'_{\mathrm{i_3}}-\mathrm{y}'_{\mathrm{i_2}}\right)^2}\tag{5.34}$$

$$l_{\mathrm{i_3}}=\sqrt{\left(\mathrm{x}_{\mathrm{i_1}}-\mathrm{x}_{\mathrm{i_3}}\right)^2+\left(\mathrm{y}_{\mathrm{i_1}}-\mathrm{y}_{\mathrm{i_3}}\right)^2},\ l'_{\mathrm{i_3}}=\sqrt{\left(\mathrm{x}'_{\mathrm{i_1}}-\mathrm{x}'_{\mathrm{i_3}}\right)^2+\left(\mathrm{y}'_{\mathrm{i_1}}-\mathrm{y}'_{\mathrm{i_3}}\right)^2}\tag{5.35}$$

$$Q_i=\cos^{-1}\left(\sqrt{\frac{l_{\mathrm{i_3}}^2-(l_{\mathrm{i_1}}^2+l_{\mathrm{i_2}}^2)}{2.l_{\mathrm{i_1}}.l_{\mathrm{i_2}}}}\right),\ Q'_j=\cos^{-1}\left(\sqrt{\frac{l'^2_{\mathrm{j_3}}-(l'^2_{\mathrm{j_1}}+l'^2_{\mathrm{j_2}})}{2.l'_{\mathrm{j_1}}.l'_{\mathrm{j_2}}}}\right)\tag{5.36}$$

5.3.1.1.3. Coincidencia de las Invarianzas de Triangulación

Basamos la decisión para la determinación de la igualdad de triángulos en las características geométricas que presenten (las Invarianzas de Triangulación), unas de las condiciones más conocidas toma en cuenta la igualdad de los lados correspondientes a ambos triángulos. Asumimos esta condición por la simpleza computacional que requiere su implementación.

Como lo que se desea es alinear conjuntos de puntos, no sólo se requiere determinar los posibles triángulos coincidentes sino también la transformación de similitud relacionada con cada par de triángulos coincidentes, puesto que estas transformaciones son las que permiten establecer el alineamiento entre conjuntos de puntos. La determinación de dicha transformación requiere resolver el sistema de ecuaciones:

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x' \\ y' \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{5.37}$$

Con respecto a las variables a, b, c, d, e y f para lo cual se cuenta con los seis pares coordenados correspondientes a los dos triángulos coincidentes, sin una pérdida de generalidad, denotamos los pares coincidentes como:

$$\left\{(x_1, y_1), (x_2, y_2), (x_3, y_3)\right\} \in \overline{\zeta}_p \tag{5.38}$$

$$\left\{(x_1', y_1'), (x_2', y_2'), (x_3', y_3')\right\} \in \overline{\zeta}_s \tag{5.39}$$

Entonces, la solución del sistema de ecuaciones viene dado por las expresiones:

$$a = \frac{(x_1.(y_3'-y_2')+x_2.(y_1'-y_3')+x_3.(y_2'-y_1'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \tag{5.40}$$

$$b = \frac{(x_1.(x_2'-x_3')+x_2.(x_3'-x_1')+x_3.(x_1'-x_2'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \tag{5.41}$$

$$c = \frac{(y_1.(y_3'-y_2')+y_2.(y_1'-y_3')+y_3.(y_2'-y_1'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \tag{5.42}$$

$$d = \frac{(y_1.(x_2'-x_3')-y_2.(x_3'-x_1')+y_3.(x_1'-x_2'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \tag{5.43}$$

$$e = -\frac{(x_1.(y_3'.x_2'-y_2'.x_3')+x_2.(x_3'.y_1'-x_1'.y_3')+x_3.(x_1'.y_2'-x_2'.y_1'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \tag{5.44}$$

$$f=-\frac{(y_1.(y_3'.x_2'-y_2'.x_3')+y_2.(x_3'.y_1'-x_1'.y_3')+y_3.(x_1'.y_2'-x_2'.y_1'))}{(x_1'.(y_3'-y_2')+x_2'.(y_1'-y_3')+x_3'.(y_2'-y_1'))} \quad (5.45)$$

Los pasos necesarios para llevar a cabo la comparación de todos los posibles triángulos y la determinación de sus respectivas transformaciones de similitud son:

1. Se calcula la suma de las diferencias de los elementos de una tripleta de ξ_P contra todas las tripletas de ξ_S, obteniéndose una lista de índices de similitud entre triángulos.
2. Se determina el mínimo valor de la lista índices de similitud entre triángulos, si dicho índice es menor que un valor umbral (9 en nuestro caso), entonces referencia a aquella tripleta de ξ_S que se asemeja a la tripleta de ξ_P en cuestión, en caso contrario no existirá ninguna tripleta de ξ_S que se asemeje a la tripleta de ξ_P en cuestión.
3. Los pasos 1 y 2 se ejecutan hasta haber analizado todas las tripletas de ξ_P, obteniéndose aquellos triángulos de ξ_P y ξ_S coincidentes.
4. Para cada par de triángulos coincidentes, utilizando sus correspondientes tripletas de puntos $\overline{\zeta}_P$ y $\overline{\zeta}_S$ se determina su transformación de similitud por medio de las ecuaciones (5.40), (5.41), (5.42), (5.43), (5.44) y (5.45).
5. La transformación de similitud determinada es validada a través del uso de la correlación local aplicada a las vecindades locales centradas en los respectivos vértices de cada triángulo, si las correlaciones son menores que un valor umbral (0.5 en nuestro caso) se considera que la transformación es válida.

En la figura 5.9 se muestra un esquema del proceso de correlación local entre vecindades centradas en los vértices de cada par de triángulos coincidentes.

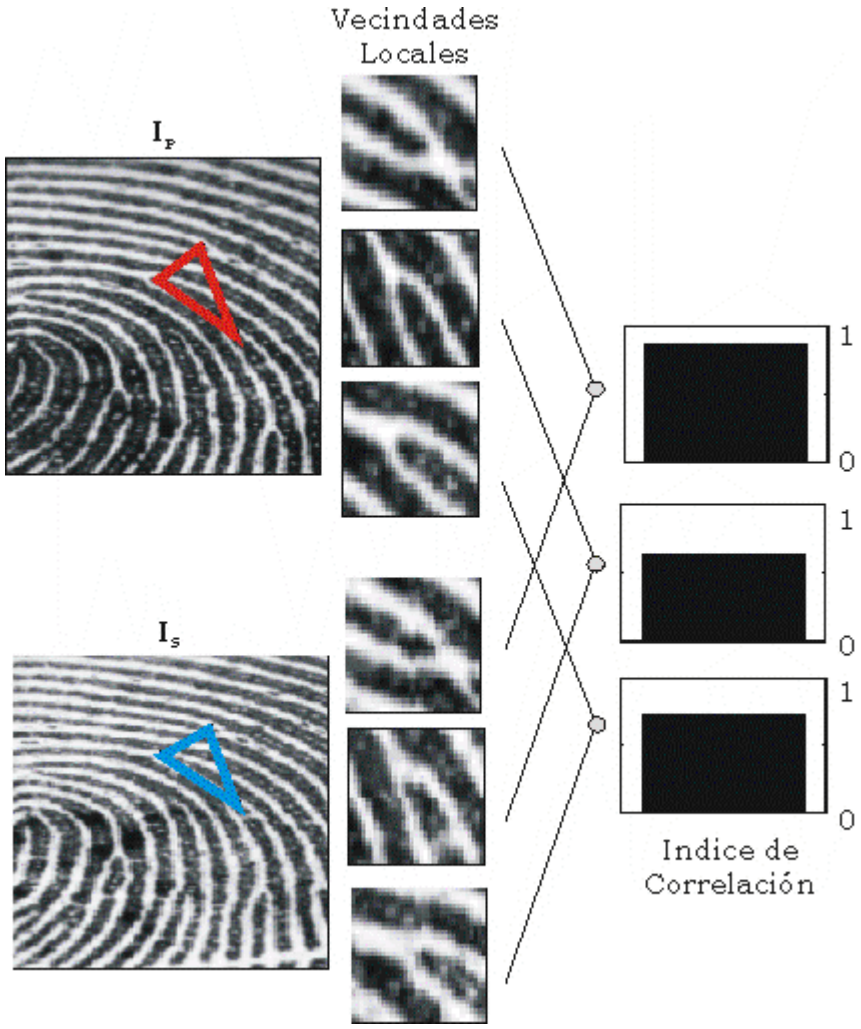


Figura 5.9 Esquema de la correlación local.

5.3.1.1.4. Determinación del índice de coincidencia

El paso para la determinación del índice de coincidencia entre $\mathbf{I}_P$ e $\mathbf{I}_S$ está basado en el conteo de minucias apareadas luego de haber realizado el alineamiento de los dos conjuntos de minucias usando las transformaciones de similitud calculadas en el paso previo. Los pasos necesarios para realizar el alineamiento y apareamiento son:

1. Cada transformación de similitud es aplicada a las posiciones de las minucias de υ_S, obteniéndose de esta manera el nuevo conjunto de minucias alineadas υ'_S.

$$\upsilon'_S = \left\{m''_1, m''_2, \ldots, m''_{N_{MS}}\right\} \tag{5.46}$$

$$\begin{bmatrix} x''_j \\ y''_j \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} x'_j \\ y'_j \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} \tag{5.47}$$

2. Se determina el número de minucias apareadas entre los conjuntos de minucias υ_P y υ'_S, definiéndose el índice de coincidencia:

$$IMth = \frac{N_{APRD}}{N_{MP}} \tag{5.48}$$

Donde:

N_{MP}: Es el número de minucias de la imagen primaria.

N_{APRD}: Es el número de minucias apareadas o coincidentes

3. De todos los índices de coincidencia determinados (por cada par de triángulos coincidentes se tendrá un índice de coincidencia respectivo) el de mayor valor es el que indica el grado de similitud entre la imagen $\mathbf{I}_P$ y $\mathbf{I}_S$ en cuestión.

En la figura 5.10 se muestra la coincidencia de minucias para dos imágenes de la base de datos 2 del FVC2002.

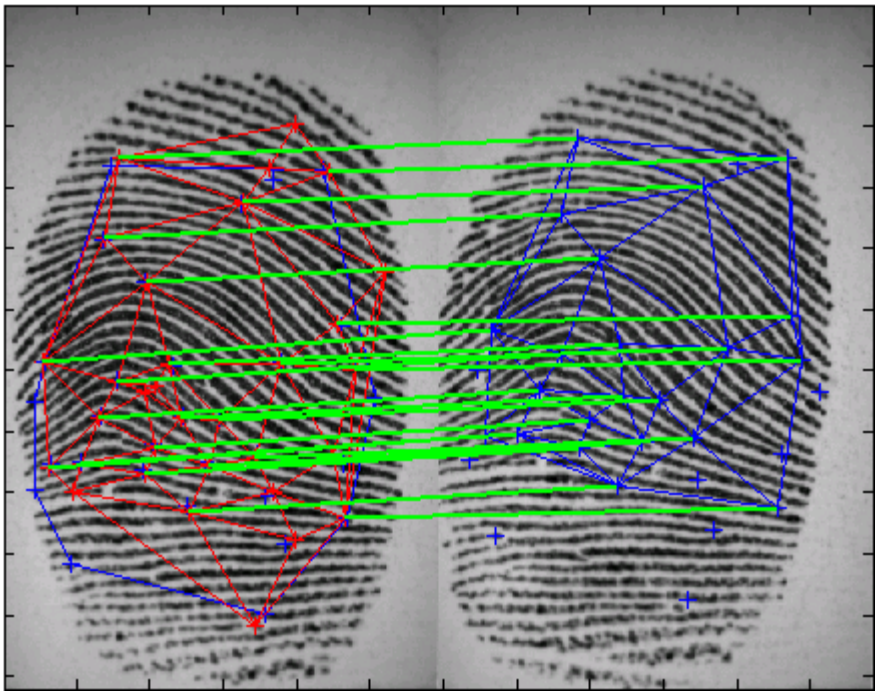


Figura 5.10 Coincidencia de las impresiones 10_5.tif y 10_2.tif de la base de datos 2 del FVC2002.

5.4. Técnicas basadas en las características de las crestas

Tres razones principales inducen a los desarrolladores de técnicas de reconocimiento por huellas dactilares a buscar otras características distintivas más allá de las minucias:

1. Extraer confiablemente las minucias de una huella dactilar de baja calidad es difícil.
2. La extracción y la coincidencia de minucias consumen mucho tiempo y costo computacional.
3. Características adicionales podrían ser usadas en conjunto con las minucias (y no como una alternativa) para incrementar la exactitud y robustez del sistema.

Las características comúnmente usadas son:

1. Tamaño y forma de la silueta externa de la huella dactilar.
2. Número, tipo y posición de las singularidades.
3. Relación espacial y atributos geométricos de las líneas de crestas [Xiao y Bian, 1986] y [Kaymaz y Mitra, 1992];
4. Características de la forma [Takeda et al, 1990] y [Ceguerra y Koprinska, 2002].
5. Información de la textura global y local.
6. Poros sudorosos [Stosz y Alyea, 1994]
7. Características fractálicas [Polikarpova, 1994]

De la lista anterior, la característica que ofrece mayores ventajas y alternativas respecto a las minucias son las basadas en la información de textura (quinto punto de la lista anterior). Las texturas son definidas por la repetición espacial de elementos básicos y son caracterizados por propiedades como escala, orientación, frecuencia, geometría, isotropía, entre otros. Las técnicas de mayor interés son las propuestas por: [Coetzee y Botha, 1993] y [Willis y Myers, 2001] quienes plantearon el análisis de textura en el dominio de Fourier, [Jain et al., 2000] desarrolla-

ron una técnica de análisis de textura a través de un banco de filtros de Gabor bidimensional que genera un vector de características denominado *Finger Code*.

La obtención del *Finger Code* ha sido estudiada en detalle en la sección 4.4.1. A continuación detallamos la técnica de coincidencia utilizada para este tipo de representación de la impresión dactilar.

5.4.1. Coincidencia basada en el FingerCode

La técnica de coincidencia descrita a continuación está basada en el trabajo de [Jain et al., 2000] y tiene como característica principal la simplicidad de su implementación (ideal para sistemas embebidos o autónomos). Los dos principales pasos de esta técnica son (ver figura 5.11):

1. **Extracción de Características**. Se determinan las características que tendrá la sectorización y el banco de filtros de Gabor bidimensional para la obtención de los *Finger Code*.
2. **Coincidencia a través de la distancia Euclidiana**. Se comparan los *Finger Code* de la imagen primaria y secundaria a través del cálculo de la distancia Euclidiana.

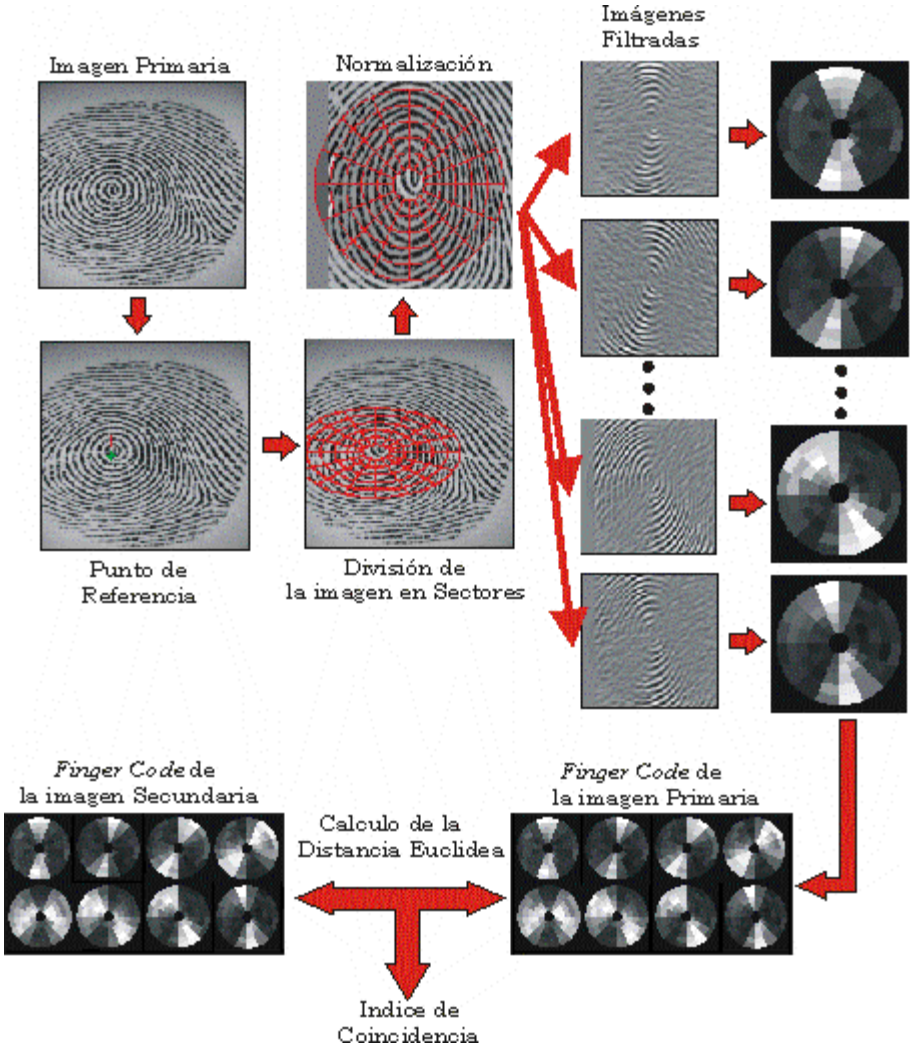


Figura 5.11 Diagrama del sistema de Coincidencia para el *Finger Code* planteado por [Jain et al. 2000].

5.4.1.1. Extracción de características

Hemos usado la representación basada en el banco de filtros de Gabor bidimensional con los parámetros que a continuación se indican.

En nuestros experimentos iniciales con la base de datos 2 del FVC2002 (el tamaño de la imagen es 560x296 píxeles, escaneada a 569 dpi), consideramos cinco bandas concéntricas (B = 5). Cada banda es de veinte píxeles de ancho (b= 20), y esta segmentada en dieciséis sectores (k = 16). Así, tenemos un total de 80 (=16 x 5) sectores (de S_0 a S_{79}) y la región de interés estará conformada por un círculo de radio 120 píxeles, centrados en el punto de referencia. Ochenta carac-

terísticas para cada una de las ocho imágenes filtradas proporcionan un total de 640 (=80x8) características por cada impresión dactilar. Cada característica puede ser cuantizada en 256 valores requiriendo un byte de almacenamiento, por lo que para almacenar todo el vector de características se necesitarán sólo 640 bytes de almacenamiento. Un punto a considerar es que estos parámetros de la sectorización dependen del tamaño y la resolución de la imagen.

En la figura 5.12 se muestran los *Finger Codes* (como imágenes en escala de grises) para ocho impresiones de la misma huella. El nivel gris de cada sector (perteneciente a un disco) representa el valor de la característica. Estas impresiones dactilares se diferencian una de otra en la traslación, rotación y deformaciones no lineales de las intensidades de la imagen. A pesar de eso vemos que las representaciones para las ocho impresiones son visualmente muy parecidas. Sin embargo, los valores característicos no son exactamente los mismos.

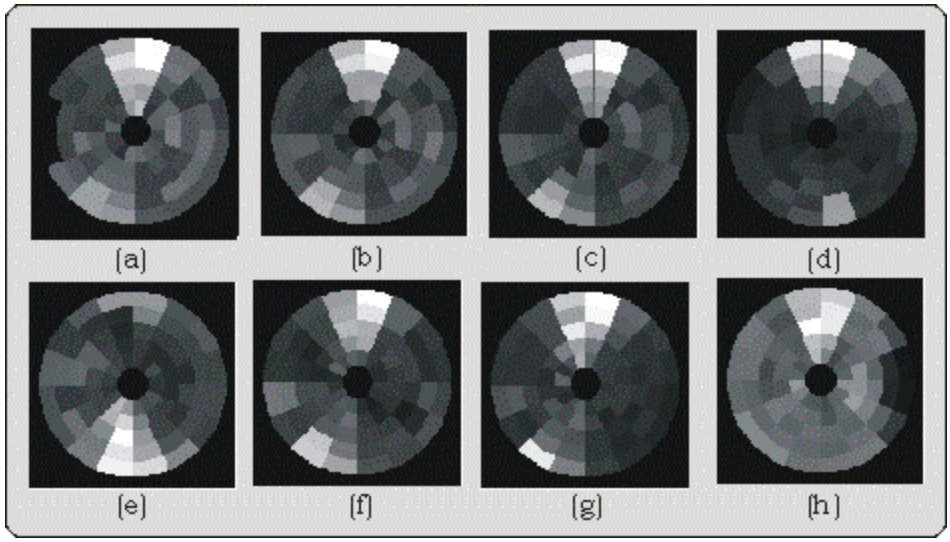


Figura 5.12 Ejemplos de *Finger Codes* de 640 características correspondientes a ocho diferentes impresiones de la misma huella (a) 50_1.tif; (b) 50_2.tif; (c) 50_3.tif; (d) 50_4.tif; (e) 50_5.tif; (f) 50_6.tif; (g) 50_7.tif; (h) 50_8.tif. Imágenes de la base de datos 2 del FVC2002.

5.4.1.2. Coincidencia a través de la distancia Euclidiana

La coincidencia de las huellas dactilares está basada en la determinación de la distancia Euclidiana entre los *Finger Code* correspondientes a la imagen primaria y secundaria. La invarianza a la traslación en el *Finger Code* es obtenida a través del punto de referencia. Sin embargo, los *Finger Code* no son invariantes a la rotación. La aproximación para la invarianza a la rotación es lograda por la rotación cíclica del *Finger Code*. Un único paso en la rotación cíclica es descrito por las ecuaciones (5.49), (5.50) y (5.51) que corresponde a un vector de características en caso de que la imagen sea rotada 22.5^o. Una rotación por R pasos corresponde a una rotación de $Rx22.5^o$ de la imagen (ver la figura 5.13). El *Finger Code* obtenido después de R pasos de rotación es dado por:

$$V_{i\theta}^{R} = V_{i'\theta'} \tag{5.49}$$

$$i' = (i + k - R)\%k + \lfloor (i/k) \rfloor k \tag{5.50}$$

$$\theta' = (\theta + 180^o + 22.5^o \, x(-R))\%180^o \tag{5.51}$$

Donde $V_{i\theta}^{R}$ es el *Finger Code* rotado, $V_{i'\theta'}$ es el *Finger Code* original, k(=16) es el número de sectores en una banda,

$$i \in [0,1,2,\cdots,79], \text{ y } \theta \in [0^o, 22.5^o, 45^o, 67.5^o, 90^o, 112.5^o, 135^o, 157.5^o]$$

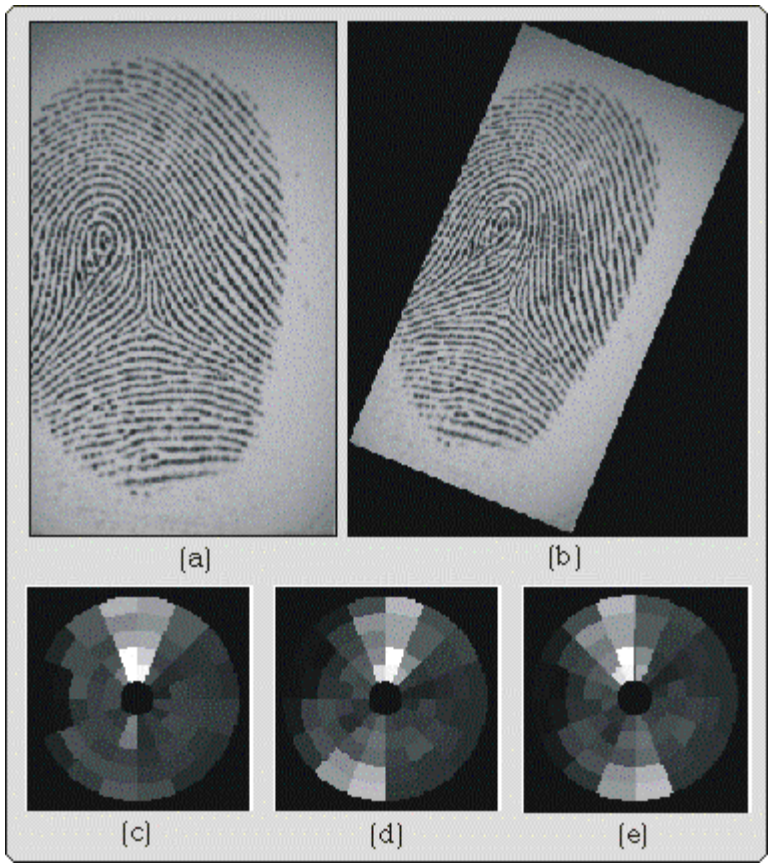


Figura 5.13 (a) Imagen original; (b) imagen rotada -22.5º; (c) *Finger Code* correspondiente a la imagen original filtrada con la máscara de Gabor sintonizada 0º ; (d) *Finger Code* correspondiente a la imagen rotada -22.5º filtrada con la máscara de Gabor sintonizada 0º ; (e) *Finger Code* rotado -22.5º correspondiente a la imagen rotada. Imagen 72_2.tif de la base de datos 2 del FVC2002.

Para cada impresión de la base de datos, almacenamos 5 plantillas correspondientes a las cinco rotaciones del *Finger Code*: $V_{i\theta}^{-2}, V_{i\theta}^{-1}, V_{i\theta}^{0}, V_{i\theta}^{1}, V_{i\theta}^{2}$. Usamos los siguientes valores para el parámetro R = {-2,-1,0,1,2} debido a que algunas imágenes en la base de datos 2 del FVC2002 tienen un marcada rotación. El *Finger Code* de entrada es coincidido con las cinco plantillas almacenadas en la base de datos para obtener cinco diferentes índices de coincidencia. El mínimo de estos cinco índices de coincidencia corresponde al mejor alineamiento de la impresión dactilar primaria con las impresiones dactilares secundarias. Puesto que una única rotación cíclica corresponde a una rotación de 22.5º en la imagen original, podemos únicamente generar aquellas representaciones de la huella dactilar, las cua-

les son múltiplos de 22.5º. Debido a la naturaleza de la sectorización, el *Finger Code* es invariante únicamente a pequeñas perturbaciones que sean menos de ±11.25º. Por consiguiente, en el momento de inscribir al usuario será necesario generar otro vector de características para cada impresión dactilar correspondiente a una rotación de 11.25º de la imagen, obteniéndose cinco plantillas adicionales a las ya obtenidas anteriormente, por lo que se tendrán diez plantillas por cada impresión dactilar perteneciente a la base de datos. Estas diez plantillas corresponden a todas las rotaciones de la impresión dactilar en múltiplos de 11.25º. El índice de coincidencia final es tomado como el mínimo de los diez índices obtenidos. Este mínimo índice corresponde al mejor alineamiento de las dos huellas dactilares.

Puesto que la generación y almacenamiento de los *Finger Codes* para las imágenes de la base de datos es un proceso *off-line* la técnica de coincidencia es extremadamente rápida, pues el tiempo de verificación todavía depende del tiempo tomado para generar un *Finger code* para la imagen primaria.

5.5. Resultados Experimentales

5.5.1. Bases de Datos

A lo largo de este capítulo tres técnicas para la coincidencia de impresiones dactilares han sido examinadas y sus detalles de implementación han sido explicados. Sin embargo, no se ha determinado cuál es la mejor técnica para la coincidencia de impresiones dactilares. Hay dos razones principales por las cuales es difícil evaluar el desempeño de las técnicas de coincidencia, siendo éstas:

1. La ejecución de la tarea de reconocimiento involucra un compromiso entre las diferentes medidas de rendimiento: exactitud, eficiencia, tamaño de la imagen y así sucesivamente. Diferentes aplicaciones tienen diferentes requerimientos.
2. La mayoría de trabajos científicos publicados incluyen resultados experimentales llevados a cabo en una base de datos propia usando diferentes protocolos, los cuales no son usualmente compartidos con la

comunidad investigadora. Esto hace difícil de comparar resultados para diferentes técnicas.

Antes de las competencias FVC2000 [Maio et al., 2002a] y FVC2002 [Maio et al., 2002b], la única base de datos de impresiones dactilares de dominio público era la brindada por el Instituto Nacional de Estándares y Tecnología (*National Institute of Standards and Technology* - NIST); a pesar que esta base de datos constituye un excelente punto de referencia para el desarrollo de los AFIS [Shen y Khanna, 1994] y para los estudios de clasificación de huellas dactilares, está no es la más adecuada para la evaluación de técnicas de coincidencia basadas en imágenes capturadas *on-line*.

Las competencias FVC2000 y FVC2002 fueron organizadas con el objetivo de brindar bases de datos a cualquier investigador interesado y rastrear el rendimiento de las técnicas de coincidencia de impresiones dactilares. Afortunadamente, muchos autores ahora reportan los resultados de sus experimentos en estas bases de datos de acuerdo al protocolo propuesto; de esta manera producen resultados que pueden ser comparados con toda la comunidad científica. Se espera que ésto se vuelva una práctica común para los científicos y practicantes en este campo.

Cuatro bases de datos son brindadas por el FVC2002 (DB1, DB2, DB3 y DB4). Cada una de éstas ha capturado 110 huellas y por cada huella ha registrado ocho impresiones (880 impresiones dactilares en total). Para la prueba de las técnicas de coincidencia se seleccionó la base de datos DB2 cuyas imágenes fueron capturadas con el sensor óptico FX2000 de BiometriKa [URL1 5.5] a 569 dpi, con un tamaño de 296x560 píxeles. Las impresiones dactilares varían en calidad (ver figura A.2), aproximadamente el 80% de las impresiones son de buena calidad (de cada ocho impresiones dos son de pobre calidad), debiéndose principalmente a cortes y manchas en las crestas.

5.5.2. Medidas de Evaluación

Las medidas de evaluación de cada técnica fueron obtenidas respecto a la base de datos DB2, reportándose los siguientes indicadores de rendimiento:

Distribuciones Genuinas y Impostoras.

Curvas FMR (t)/FNMR (t).

Curva ROC(t).

EER (Tasa de Igual Error)

FMR100 (Es el FNMR mas baja para FMR<=1%)

FMR1000 (Es el FNMR mas baja para FMR<=0.1%)

ZeroFMR (Es el FNMR mas baja para FMR=0%)

ZeroFNMR (Es el FMR mas baja para FNMR=0%)

Promedio del Tiempo de Inscripción.

Promedio del Tiempo de Coincidencia.

Los procedimientos de medida para los indicadores de rendimiento de mayor importancia son descritos a continuación:

Cálculo de la Tasa de Falsa no Coincidencia FNMR (también referida como la Tasa de Falso Rechazo - FRR):

1. Cada impresión de la base de datos es comparada contra las impresiones restantes de la misma huella obteniéndose sus respectivos índices de coincidencia. Si se realiza la coincidencia de g contra h, lo simétrico (es decir, h contra g) no se ejecuta para evitar la correlación.
2. Se establece un valor umbral de decisión t_o.
3. Todos los índices de coincidencia son comparados con t_o, si algún índice es mayor implicará la presencia de un falso rechazo, contabilizándose el número de falsos rechazos presentes.
4. Los pasos (2) y (3) son llevados a cabo para diferentes valores umbrales de decisión, obteniéndose así la curva FNMR versus los valores umbrales de decisión.

El número total de pruebas genuinas (en caso de que no ocurra algún rechazo de inscripción) es 2 800 (=(7*8) /2) * 100) .

Cálculo de la Tasa de Falsa Coincidencia FMR (también conocida como Tasa de Falsa Aceptación - FAR):

1. La primera impresión de cada huella de la base de datos es comparada contra la primera impresión de las huellas restantes en la base de datos obteniéndose sus respectivos índices de coincidencia. Si se realiza la coincidencia de g contra h, lo simétrico (es decir, h contra g) no se ejecuta para evitar la correlación.
2. Se establece un valor umbral de decisión t_o.
3. Todos las índices de coincidencia son comparados con t_o , si algún índice es menor implicará la presencia de una falsa aceptación, contabilizándose el número de falsas aceptaciones presentes.
4. Los pasos (2) y (3) son llevados a cabo para diferentes valores umbrales de decisión, obteniéndose así la curva FMR versus umbrales de decisión.

El número total de pruebas impostoras (en caso de que no ocurra algún rechazo de inscripción) es 4 950 (=(99*100) /2).

Cálculo de las distribuciones Genuinas e Impostoras: La distribución Genuina/Impostora es el histograma vinculado al número de pruebas genuinas/ impostoras y se calcula a través de los siguientes pasos:

1. Se establece un rango para los posibles valores del índice de coincidencia.
2. Se determina el número de pruebas genuinas/impostoras que toman el r-ésimo valor posible del índice de coincidencia definido en el paso 1, siendo dicho valor normalizado respecto al número total de pruebas genuinas/ impostoras realizadas, obteniéndose así la probabilidad que aparezca el valor r-ésimo del índice de coincidencia.
3. Los valores de probabilidad determinados en el paso 2 son los que conforman la gráfica de distribución Genuina/Impostora.

Luego de haber calculado las distribuciones Genuinas e Impostoras se determina un indicador estadístico que cuantifica la separación entre ambas distribuciones, siendo ésta definida como la diferencia entre los valores promedios de las distribuciones genuinas e impostoras divididas por el conjunto de medidas de sus desviaciones estándares, es decir:

$$d = \frac{\left|M_{Imp} - M_{Gen}\right|}{\sqrt{(SD_{Imp}^2 + SD_{Gen}^2)/2}} \qquad (5.52)$$

Donde:

M_{Gen} y M_{Imp}: Son los valores promedios de las distribuciones Genuinas e Impostoras respectivamente.

SD_{Gen} y SD_{Imp} : Desviaciones estándares de las distribuciones Genuinas e Impostoras respectivamente.

Cálculo de la Curva de Características del Receptor ROC: Es simplemente la gráfica de la FMR versus (1-FNMR) para todos los valores umbrales de decisión con los que se evaluó la técnica en cuestión. En el caso ideal, la curva ROC es una función escalón con un cero en el origen de coordenadas. En el peor caso, el ROC es un segmento de línea con una inclinación de 45º y un punto de finalización en la tasa de falsa aceptación de cero. En la práctica la curva ROC se comporta entre estas dos situaciones extremas.

5.5.3. Resultados

De las graficas de distribuciones Genuinas e Impostoras mostradas en la figura 5.14 podemos concluir lo siguiente:

La técnica basada en las minucias presenta la mayor separabilidad estadística (d = 0.00029638) teniendo a su vez un bajo porcentaje de rechazo (11.25% de las comparaciones son rechazadas), evidenciando así, que las minucias son las características que definen con mayor precisión la unicidad de una impresión dactilar.

La técnica basada en el *Finger Code* también presenta una separabilidad estadística aceptable (d = 0.001231), pero requiere de una política de rechazo agresiva (24.89% de las comparaciones son rechazadas), entendible por la dependencia que presenta el *Finger Code* respecto al punto de referencia.

La técnica basada en la correlación local presenta la menor separabilidad estadística (d=0.16142) con un muy bajo porcentaje de rechazo (2.83% de las comparaciones son rechazadas), pudiendo implicar que, las singularidades locales (las características que definen las vecindades a correlacionar) no establecen la unicidad de una impresión dactilar, pero considerando el porcentaje de rechazo una explicación es obvia, existen índices de coincidencia en la frontera de decisión de ambas distribuciones que al no ser rechazadas son responsables del traslape entre clases.

De las graficas de tiempo mostradas en la figura 5.15 podemos concluir lo siguiente, las tres técnicas de coincidencia presentan un tiempo de inscripción o registrado muy alto, debiéndose esta demora:

Para el caso de la técnica basada en las minucias, a los algoritmos de binarización adaptativa y esqueletización de la imagen binaria.

Para el caso de la técnica basada en el *Finger Code,* a la etapa de filtrado a través del banco de filtros de Gabor Bidimensional.

Para el caso de la técnica basada en la correlación local, a la técnica de mejora desde escala de grises implementada para la eliminación de las singularidades locales no relevantes.

De la comparación de los tiempos de coincidencia es obvio las grandes ventajas que presenta la técnica basada en el *Finger Code* (0.01 s.), seguida por la técnica basada en las minucias (1.25 s.), siendo este un resultado de interés debido a la complejidad que presenta la técnica de alineamiento y validación. Por ultimo está la técnica de correlación local (15.72 s.) resultado esperado puesto que esta basado en la correlación (por cada vecindad a correlacionar una operación de convolución es realizada). Cabe destacar que los tiempos son referenciales puesto que al ser los algoritmos implementados en MATLAB no necesariamente el tiempo

de procesamiento es proporcional a la complejidad computacional del algoritmo y por tanto al tiempo real de demora.

De las graficas de desempeño mostradas en la figura 5.17 y 5.18 podemos concluir lo siguiente: La curva ROC muestra los niveles de rendimientos de las tres técnicas, apreciandose que la técnica basada en minucias presenta el mejor desempeño, por lo que es idonea para aplicaciones de alta seguridad, además tiene el mejor comportamiento según la EER (EER = 11,22%), a continuación tenemos la técnica basada en correlación (EER = 22,91%) y la técnica basada en el *Finger Code* (EER = 28,18%) teniendo como ventaja su poco tiempo de coincidencia lo que permite dos usos principales del *Finger Code* como una característica más a combinar con las otras características, o como plantilla necesaria para un sistema que realiza varías comparaciones para el establecimiento de la identidad.

Cabe destacar que el EER para las tres técnicas de coícindencia depende en gran medida de la tasa de rechazo. Las tablas 5.3 y 5.4 muestran lo expresado para las técnicas de coincidencia basadas en minucias y correlación, obteniéndose para el caso de la técnica basada en las minucias un EER igual a 6% cuando es rechazado el 23,7% de las comparaciones y para la técnica basada en correlación un EER igual a 18,19% cuando es rechazado el 24% de las comparaciones.

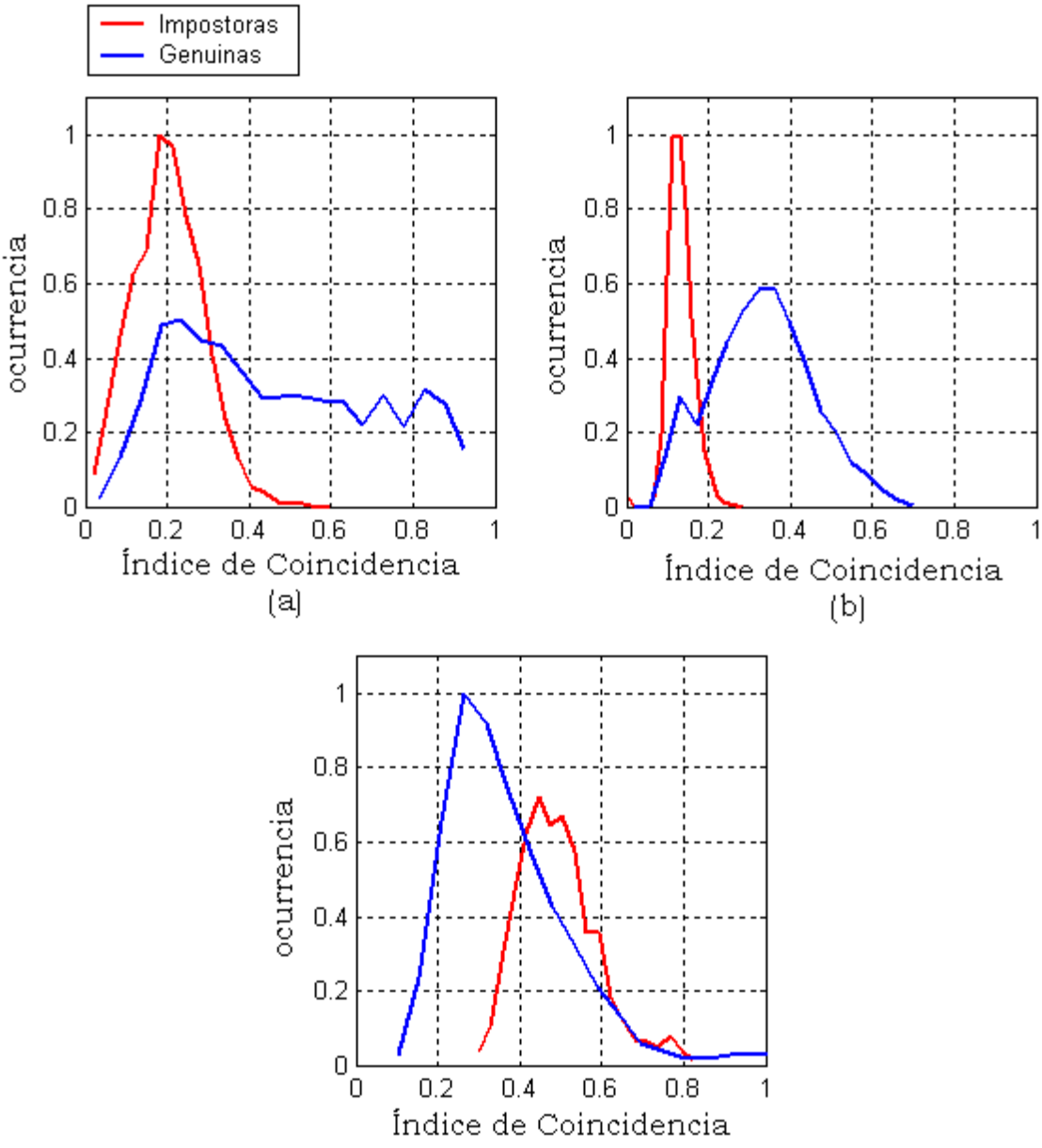


Figura 5.14 Distribuciones Genuinas e Impostoras para las tres técnicas de coincidencia (a) basada en la correlación local; (b) basada en las minucias; (c) basada en el *Finger Code* .

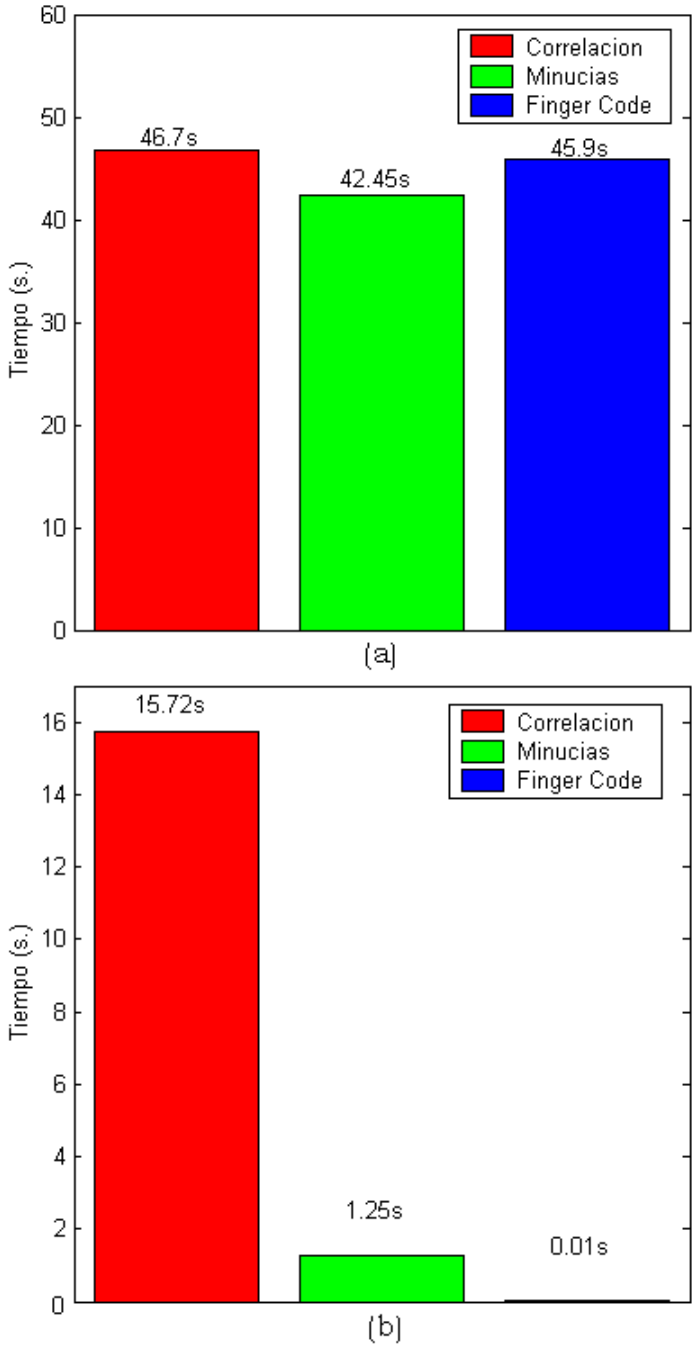


Figura 5.15 Comparación de los tiempos de procesamiento para cada una de las técnicas desarrolladas (a) tiempo de extracción de características (inscripción); (b) tiempo de coincidencia.

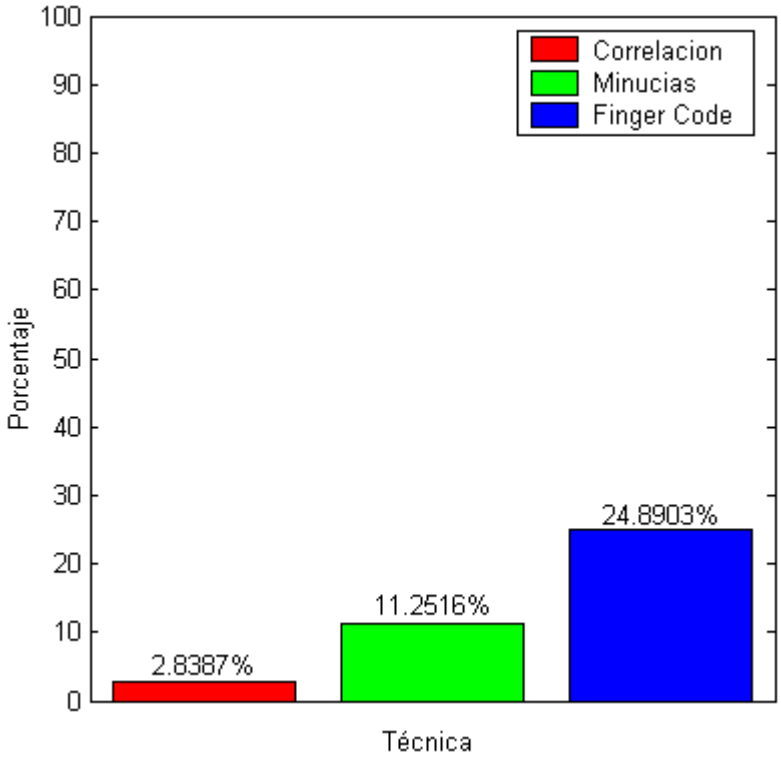


Figura 5.16 Comparación de los porcentajes de rechazo para cada una de las técnicas desarrolladas.

	Correlación	Minucias	*Finger Code*
d	0.16142	0.00029638	0.001231
TimeInsp(s)	46.7	42.45	45.9
TimeMatch(s)	15.72	1.25	0.01
Rechazo(%)	2.83	11.25	24.89

Tabla 5.1 Medidas de rendimiento para la separabilidad estadística y tiempos de procesamiento.

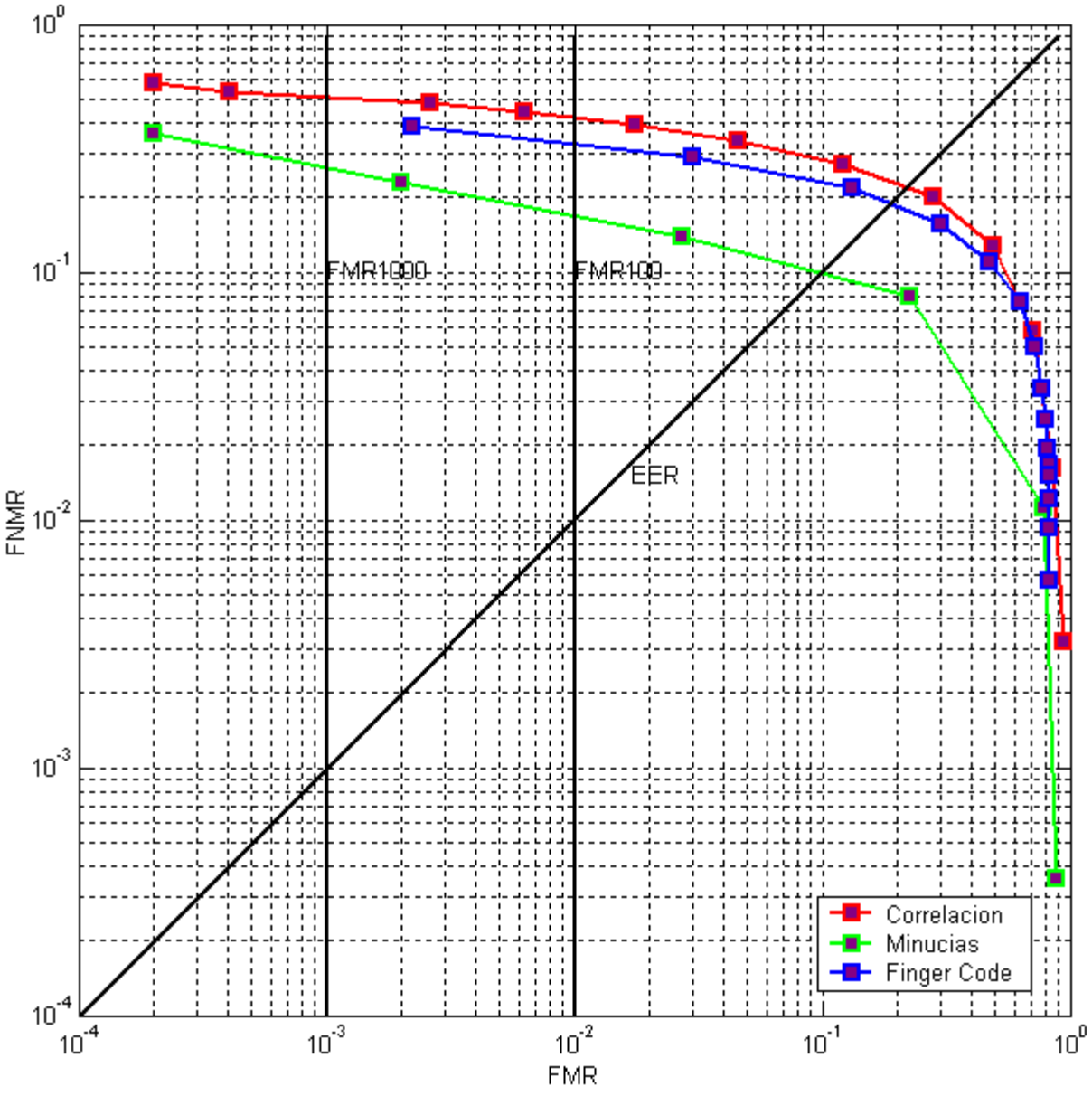


Figura 5.17 FNMR versus FMR para las tres técnicas de coincidencia.

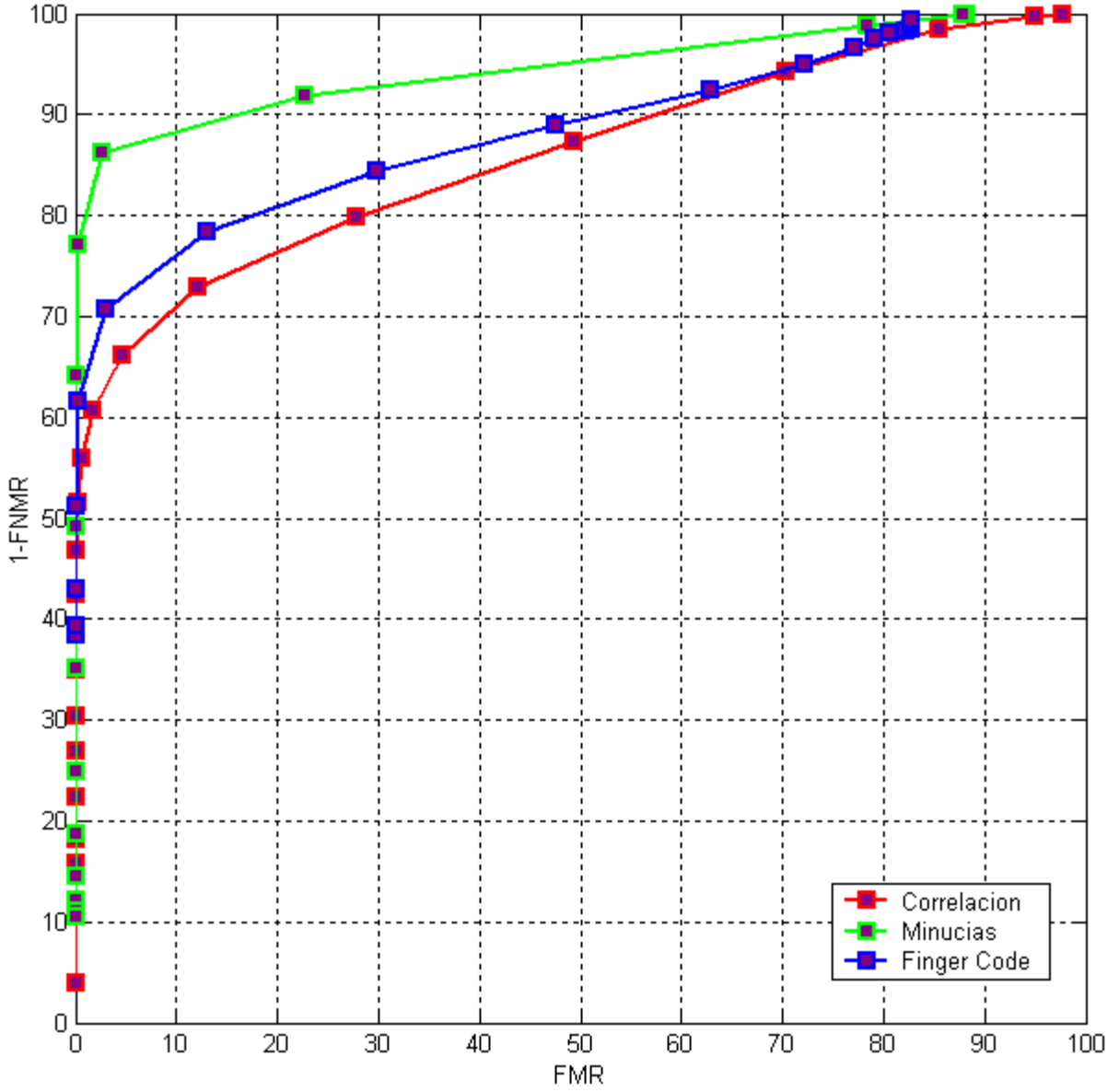


Figura 5.18 Curva de ROC comparativa para las tres técnicas de coincidencia.

	CORRELACIÓN	MINUCIAS	*FINGER CODE*
EER	22,91%	11,22%	28,18%
FMR100	43,65%	19,50%	54,33%
FMR1000	53,70%	30,20%	70,79%
ZeroFNMR	99,72%	99,54%	100,00%
ZeroFMR	60,26%	40,13%	62,25%

Tabla 5.2 Medidas de rendimiento para las tres técnicas de coincidencia.

N° Minucias	11	12	13	14	15	16	17	18	19	20	21
Rechazo (%)	11,80%	13,30%	13,79%	16,43%	17,14%	17,97%	18,58%	18,99%	20,74%	23,88%	23,70%
EER	10,00%	9,30%	8,93%	8,72%	8,31%	8,03%	7,76%	7,60%	7,24%	6,83%	6,02%

Tabla 5.3 Diferentes EER para la técnica de coincidencia basada en las minucias.

Calidad	0.1	0.15	0.2	0.25	0.3	0.35	0.4
Rechazo (%)	6,40%	7,90%	10,90%	14,00%	16,85%	20,00%	24,00%
EER	21,37%	21,28%	20,46%	19,99%	19,49%	19,05%	18,19%

Tabla 5.4 Diferentes EER para la técnica de coincidencia basada en la correlación.

5.6. Resumen

En este capítulo se expusieron los conceptos básicos de la coincidencia de impresiones dactilares, además se detallaron tres técnicas de coincidencia basadas en las singularidades locales, en las minucias y en el *Finger Code*, también se describieron las medidas de evaluación necesarias para la determinación del desempeño de cada técnica.

En la sección dedicada a la técnica de coincidencia basada en la correlación, se hace uso de la correlación local, proponiendo a las singularidades locales cuantizadas como centros geométricos de cada vecindad a comparar, obtenidas a partir de la matriz de irregularidad de la impresión dactilar. La operación de correlación es realizada a través de la técnica de correlación cruzada normalizada por ofrecer una medida estadística entre cero y uno, siendo invariante a los cambios del contraste y brillo. Esta técnica tiene como desventaja el gran costo en tiempo de procesamiento tanto en la extracción de características como en la fase de coincidencia.

En la sección dedicada a la técnica de coincidencia basada en las minucias, hemos propuesto un algoritmo de coincidencia que alinea las minucias a través de la triangulación de Delaunay, utilizando como Invarianzas de Triangulación a las

longitudes de los lados de cada triángulo de Delaunay, incluyendo además una etapa de validación del alineamiento a través de la correlación por vecindades. Como lo indican las figuras 5.17 y 5.18 es la mejor de las tres técnicas, haciéndola la más adecuada para aplicaciones de alta seguridad y sobre todo el costo de tiempo de coincidencia es aceptable.

En la sección dedicada a la coincidencia basada en el *Finger Code*, se expusieron los pasos necesarios de la técnica de coincidencia planteada por [Jain et al., 2000], y tal como lo indica la figura 5.15 tiene el menor tiempo de coincidencia implicando una menor complejidad computacional frente a las demás técnicas, permitiendo usarla como una característica más a combinar con las otras características, o como plantilla necesaria para un sistema que realiza varías comparaciones para el establecimiento de la identidad.

Capítulo VI
Toolbox para el Reconocimiento por Huellas Dactilares (TReHD)

6.1. Introducción

En los capítulos anteriores se han detallado las técnicas algorítmicas necesarias para cada una de las correspondientes etapas que involucra el reconocimiento de huellas dactilares, en la práctica esto nos a conllevado al diseño de una herramienta computacional que permita probar los algoritmos y técnicas desarrolladas. Por ser una aplicación vinculada al procesamiento de imágenes, cuestiones como la captura de los datos y su normalización son puntos de vital importancia. Diferentes bibliotecas de funciones para lenguajes estándares, permiten realizar dichas labores sin complicaciones, pero presentando como inconvenientes la alta curva de aprendizaje, la dificultad en la depuración de las aplicaciones realizadas y en visualizar gráficamente el comportamiento estadístico de los datos. Un software que compensa en gran medida los inconvenientes anteriormente mensionados es el entorno MATLAB (*Matrix Laboratory*) desarrollado por MathWorks, el cual se podría definir como un entorno de computación y desarrollo de aplicaciones totalmente integrado, orientado para llevar a cabo proyectos en donde se encuentren implicados elevados cálculos matemáticos y la visualización gráfica de los mismos. MATLAB integra el análisis numérico, cálculo matricial, procesamiento de señales y visualización gráfica, entre muchas otras herramientas; dispone también en la actualidad de un amplio abanico de programas de apoyo especializados, denominados *Toolbox*, que extienden significativamente el número de funciones incorporadas en el programa principal. Estos *Toolbox* cubren en la actualidad prácticamente casi todas las áreas principales en el mundo de la ingeniería y la simulación, destacando entre ellos el *Toolbox* de procesamiento de señales e imágenes, estadística, redes neurales, lógica difusa, control, entre otros. En las siguientes secciones detallaremos sobre la biblioteca de funciones

desarrollada sobre MATLAB especializada para el reconocimiento de personas por sus huellas dactilares.

6.2. Metodología de programación

En un inicio se realizó un análisis del lenguaje de programación a utilizar, seleccionándose el lenguaje C++ bajo el estándar ANSI/ISO C++, a través del entorno de desarrollo integrado (*Integrated Development Environment* - IDE) de Microsoft VC++ 6.0, elaborándose una biblioteca de funciones especializada (librería) para el procesamiento de imágenes, que denominamos *JcImage* basada en las librerías *Surfwave* [Adler y Rao, 1996] y *CxImage* [URL 6.1], además vinculada a la librería IPL (*Image Processing Library*) de Intel [URL 6.2]. El diseño de esta nueva librería constó de tres fases: (1) la construcción de las clases para la manipulación de matrices; (2) la elaboración de las funciones para la lectura y escritura de archivos de imágenes; y (3) la elaboración de las funciones para el procesamiento de imágenes. Con el fin de implementar aplicaciones visuales sofisticadas y por la optimizada administración de la memoria dinámica de la PC, se migró al IDE CBuilder 5.0 de Borland. A pesar de haber llegado hasta la tercera fase de diseño, con un consumo de siete meses de programación (a la fecha actual la librería consta de mas de 5000 líneas de código) se desistió de tal labor por el tiempo de desarrollo y depuración que conllevan estos algoritmos (tomar en cuenta que en este trabajo se utilizan gran cantidad de técnicas relacionadas con el análisis y el procesamiento de datos, como son las redes neurales, cuantización vectorial, triangulación de puntos, entre otros). Tomando en cuenta lo aprendido se seleccionó el software MATLAB que compensa en gran medida las desventajas anteriormente descritas.

La construcción de una librería en MATLAB comúnmente denominada *toolbox*, fue realizada en un principio en la versión 5.3, luego se pasó a la versión 6.5, siendo esta la versión utilizada hasta la fecha, su desarrollo constó de las siguientes fases: (1) elaboración de las funciones de soporte general; (2) elaboración de las funciones para el análisis, extracción de características y coincidencia impresiones dactilares; (3) elaboración de las funciones para la evaluación de las técnicas de extracción de características y coincidencia de impresiones dactilares;

(4) elaboración de las aplicaciones a través de una interfaz gráfica de usuario (*Graphical User Interface* - GUI). Cada una de estas fases implicó varios meses de desarrollo tanto para la implementación como la depuración, llegando a crearse más de 200 funciones, que en conjunto decidimos denominarla "*Toolbox* para el Reconocimiento por Huellas Dactilares – TReHD". Un punto importante a remarcar es que todas las graficas y los resultados estadísticos presentados en este trabajo han sido obtenidos utilizando el TReHD. Veamos a continuación una descripción detallada de esta nueva herramienta computacional.

6.3. TReHD

El TReHD es una colección de funciones implementadas en MATLAB especializadas para procesamiento y reconocimiento por huellas dactilares. El *toolbox* soporta un amplio rango de operaciones tales como: Análisis, Extracción de Características y Coincidencia de Impresiones Dactilares a través de varias técnicas. No es un *Toolbox* aislado, pues trabaja con los *toolbox* de *Statistics*, *Neural Network*, *Image Processing* y *Signal Processing*. Fue diseñado con una estructura abierta, por lo que es viable extender las capacidades del TReHD al permitir a los futuros usuarios implementar sus propios algoritmos utilizando en ellos las funciones del TReHD así como modificar las funciones existentes.

El toolbox provee dos categorías de herramientas:

Funciones de líneas de comandos: Incluye a aquellas funciones llamadas desde la línea de comandos de MATLAB o de cualquier aplicación. Muchas de estas funciones son archivos m de MATLAB.

Funciones gráficas interactivas: Provee un número de herramientas interactivas que permiten manipular muchas de las funciones a través de una GUI.

A continuación veremos un listado de todas las funciones que conforman del *Toolbox*.

6.4. Funciones Base

Estructuras básicas	
FUNCIÓN	DESCRIPCIÓN
FingerPrint_Estruct.m	Retorna una estructura que contiene la información vinculada a la impresión dactilar.
ConstructorEstructura_Dim.m	Retorna una estructura que contiene los índices de cada bloque en que se ha sectorizado la imagen.

Aplicaciones básicas	
FUNCIÓN	DESCRIPCIÓN
setpath.m	Añade la ruta del TReHD a MATLAB.
baseImagenesTotal.m	Devuelve el nombre, clase e índice de calidad de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarRegiones.m	Aplicación que determina la Imagen de Regiones de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarBinarias.m	Aplicación que determina la imagen binaria de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarCoreDelta.m	Aplicación que determina las posiciones de los *Delta* y *Core* de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarMinucias.m	Aplicación que determina las minucias de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarSingularCoh.m	Aplicación que determina las singularidades de la matriz de coherencia de todas las imágenes de la base de datos DB2 del FVC2002.
prueba_GrabarFINGERCODE.m	Aplicación que determina el *Finger Code* de todas las imágenes de la base de datos DB2 del FVC2002.

6.5. Funciones para el análisis de la impresión dactilar

6.5.1. Funciones para la determinación de la Imagen de Orientación

6.5.1.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
PromedioGradienteVecindad.m	Obtiene la orientación dominante a través del promedio cuadrado de los Vectores Gradiente.
AnalisisComponente_SVD.m	Obtiene la orientación dominante a través de la Descomposición en Valores Singulares.
FiltradoGradienteVecindad.m	Suaviza la Imagen de Orientación a través de la técnica de los componentes sinusoidales.
ConsistenciaVecindad.m	Determina el nivel de consistencia en una vecindad de la Imagen de Orientación.
EstimationDiffusion.m	Suaviza la Imagen de Orientación a través de la técnica de difusión.
FingerPrint_gradiente.m	Determina la Imagen de Orientación.
FingerPrint_GradienteCentro.m	Determina la Imagen de Orientación pero la función esta optimizada para el cálculo del punto central de la imagen.
prueba_CampoDireccional.m	Aplicación

6.5.1.2. Funciones de Soporte

FUNCIÓN	DESCRIPCIÓN
DibujarGradiente.m DibujarGradiente_2.m	Dibuja la Imagen de Orientación superpuesta a la imagen original.
DibujarSuperficieGradiente.m	Dibuja la Imagen de Orientacion como una superficie de puntos.

6.5.2. Funciones para la determinación de la Imagen de Frecuencias

6.5.2.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
Fingerprint_DensidadFrecuencia.m	Calcula la Imagen de Frecuencias.
prueba_DensidadFrecuencia.m	Aplicación

6.5.3. Funciones para la determinación de la Imagen de Regiones

6.5.3.1. Funciones Principales

Imagen de Regiones a nivel de píxeles.	
FUNCIÓN	DESCRIPCIÓN
Planos_segmentacion.m	Determina los vectores de características asociados a los píxeles de la imagen que permiten la clasificación a nivel de píxeles.
FingerPrint_ImagenRegiones.m	Determina la Imagen de Regiones a nivel de píxeles.
procesamientoMorfologico.m	Mejora la Imagen de Regiones obtenida a nivel de píxeles a través del uso de procesamiento morfológico.
prueba_RegPixeles.m	Aplicación que determina la Imagen de Regiones a nivel de píxeles.

Redes Neurales	
FUNCIÓN	DESCRIPCIÓN
prueba_condensacion_SOM.m prueba_condensacion_SOM_2.m	Aplicaciones que realizan la condensación de las observaciones del conjunto de entrenamiento.
Prueba_entr_MLP.m	Aplicación para el entrenamiento de un *Perceptron* bajo la regla de aprendizaje clásica.
Prueba_entr_FMLP.m	Aplicación para el entrenamiento de un *Perceptron* bajo la regla de aprendizaje *Fuzzy Perceptron.*
prueba_sim_MLP.m	Aplicación para el entrenamiento y simulación de un *Perceptron* bajo la regla de aprendizaje clásica.
prueba_sim_FMLP.m	Aplicación para el entrenamiento y simulación de un *perceptron* bajo la regla de aprendizaje *Fuzzy Perceptron*.
prueba_MLPVsFMLP.m	Aplicación que compara el desempeño del *perceptron* bajo las dos reglas de aprendizaje.

Imagen de Regiones a nivel de bloques.	
FUNCIÓN	DESCRIPCIÓN
ExtractCart_RathaMaio.m	Extrae la magnitud promedio del Vector Gradiente y la varianza de una vecindad orientada para todos los bloques de la imagen.
FingerPrint_RegBloques.m	Determina la Imagen de Regiones a nivel de bloques.
ObtenerCentroidesRathaMaio.m	Obtiene los centroides representativos de cada clase, útiles para la clasificación a nivel de bloques a través de un clasificador k-NN.
prueba_RegBloques.m	Aplicación que determina la Imagen de Regiones a nivel de bloques.
prueba_BloquesMatCofs.m	Aplicación para la determinación de la Imagen de Regiones a nivel de bloques y generación de la Matriz de Confusión.

Indice de Calidad	
FUNCIÓN	DESCRIPCIÓN
MedidaIndiceCalidad.m	Determina el índice de calidad de una imagen.
prueba_IndiceCalidad.m	Ejemplo de la determinación del índice de calidad de una imagen.

6.5.3.2. Funciones de Soporte

FUNCIÓN	DESCRIPCIÓN
MediaVarianza.m	Determina el valor promedio y la varianza para cada píxel de la imagen.
Coherencia.m	Calcula la coherencia vinculada a los Vectores Gradiente de la imagen.
Prueba_SelCaract_EXPERTO.m	Aplicación que permite validar la Imagen de Regiones obtenida manualmente.
prueba_Generar_SampleTraining.m	Aplicación que obtiene las observaciones del conjunto de entrenamiento para la clasificación a nivel de píxeles.

6.6. Funciones para la extracción de características de la impresión dactilar

6.6.1. Funciones para la extracción de los puntos singulares de la impresión dactilar

6.6.1.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
FingerPrint_Poincare.m	Determina los *Delta* y *Core* presentes en una impresión dactilar a través del índice de Poincaré.
FingerPrint_CentroPoincaire.m	Determina el punto de referencia de la impresión dactilar a través de la técnica basada en el índice de Poincaré.
Fingerprint_SingularidadesDeriv.m	Determina el punto de referencia de la impresión dactilar a través de la técnica basada en los componentes sinusoidales de la Imagen de Orientación.
DeteccionManualCentro.m	Permite detectar las posiciones de los *Delta* y *Core* manualmente.
prueba_PuntoCentralFinal.m	Aplicación que muestra la determinación del punto de referencia a través de la técnica basada en el índice de Poincaré.
prueba_SingularDeriv.m	Aplicación que muestra la determinación del punto de referencia a través de la técnica basada en los componentes sinusoidales de la Imagen de Orientación.
MedidaVariacionCentroCalidad.m	Aplicación que permite medir la variación del punto de referencia respecto a la calidad de la imagen.
prueba_RotarImagen.m	Aplicación que permite medir la variación del punto de referencia respecto a la rotación de la imagen.

6.6.1.2. Funciones de Soporte

FUNCIÓN	DESCRIPCIÓN
curva_digital.m	Genera los índices de la curva sobre la vecindad en la cual se calculará en índice de Poincaré.
PosicionesDeltaCore.m	Determina las posibles posiciones de los *Delta* y *Core*.
DeltaCoreValido.m	Valida los posibles *Delta* y *Core*.
AgruparSingularidades_LQV.m	Realiza la cuantización vectorial para datos bidimensionales.
MatrixCovarianza.m	Calcula la matriz de covarianza de una matriz.
dibujar_DeltaCore.m	Dibuja los puntos *Delta* y *Core* superpuestos a la imagen original.

6.6.2. Funciones para la extracción de las minucias

6.6.2.1. Funciones Principales

Mejora	
FUNCIÓN	DESCRIPCIÓN
Fingerprint_Contraste.m	Varía el contraste de una imagen en escala de grises.
Fingerprint_Equalizado.m	Realiza la mejora de la impresión dactilar a través del aumento de contraste y la ecualización del histograma sobre aquellos píxeles pertenecientes al primer plano de la imagen.
Fingerprint_MejoraFFT2.m	Realiza la mejora de una imagen en escala de grises utilizando la magnitud de la transformada de Fourier bidimensional de la imagen.

Binarización	
FUNCIÓN	DESCRIPCIÓN
FingerPrint_Binarizacion.m	Binariza una imagen en escala de grises utilizando un umbral adaptivo.
Fingerprint_BinDireccional.m	Binariza una imagen en escala de grises utilizando la segunda derivada direccional de la imagen.
Fingerprint_esqueletizacion.m	Genera la imagen esqueletizada de una imagen binaria.
prueba_BinDireccional_Final.m	Aplicación para la binarización direccional de la imagen.

Pre Proceso	
FUNCIÓN	DESCRIPCIÓN
conectado_4.m	Determina los píxeles 4-conectados de una imagen binaria.
eliminacion_bug.m	Eliminación de píxeles *bug* presentes en la imagen esqueletizada.
Eliminacion_Esquinas.m	Eliminación de píxeles.

Detección de Minucias	
FUNCIÓN	DESCRIPCIÓN
Puntos_Singulares.m	Calcula el número de cruces por cero para todo píxel de la imagen esqueletizada.
Hallando_listas.m	Halla las listas con las posiciones de las minucias de finalización y bifurcación.
Comprobando_Condiciones.m	Determina si un punto de una cresta adelgazada ya fue etiquetado como la posición de alguna minucia.
Vias_Finalizacion.m	Genera una estructura que contiene las rutas, orientación y la posición de cada minucia de finalización.
Vias_Bifurcacion.m	Genera una estructura que contiene las rutas, orientación y la posición de cada minucia de bifurcación.
Vias_Bridges.m	Genera una estructura que contiene las rutas de todos los posibles puentes y además las minucias que las conforman.
Fingerprint_DetecMinu.m	Determina las posiciones y orientación de las minucias de finalización y bifurcación a partir de la imagen esqueletizada de la impresión dactilar.
DeteccionManual.m	Permite detectar las posiciones de las minucias manualmente.
CalculoIndiceVirtud.m	Determina el Índice de Virtud.
MedidaIndiceVirtud_GI.m	Aplicación para el cálculo del Índice de Virtud sobre varios conjuntos de minucias.
MedidaBaseline.m	Aplicación para la obtención de la distribución de Baseline.
prueba_DeteccionMinucias.m	Aplicación para la detección de minucias.

Post Procesamiento	
FUNCIÓN	DESCRIPCIÓN
Eliminacion_MinuBorde.m	Elimina todas aquellas minucias que se encuentran en el segundo plano de la Imagen de Regiones.
Eliminacion_Spurs.m	Elimina las minucias de bifurcación y finalización unidas por rutas muy pequeñas.
Eliminacion_ShortBreaks.m	Elimina las minucias de finalización que son parte de crestas rotas.
Eliminacion_Island.m	Elimina las minucias de finalización cuya ruta es muy pequeña.
Eliminacion_Bridges.m	Elimina las minucias que conforman los puentes.
Eliminacion_Borders.m	Elimina minucias ubicadas en el contorno formado por el primer y segundo plano de la imagen de regiones.
BorradoMinucias.m	Borra todas las estructuras vinculadas a las minucias eliminadas.

6.6.2.2. Funciones de Soporte

Mejora	
FUNCIÓN	DESCRIPCIÓN
Contraste.m	Varía el contraste de una imagen en escala de grises.
Intensidad.m	Varía la intensidad de una imagen en escala de grises.
IntensidadContraste.m	Varía el contraste y la intensidad de una imagen en escala de grises.
Im_hist.m	Determina el histograma de una imagen.
Im_Equal.m	Realiza una mejora de una imagen en escala de grises a través del ecualizado del histograma.
prueba_MejoraFFT2_Final.m	Aplicación para la mejora de la imagen a través de la FFT2
sigmoidea.m	Genera la función sigmoidea.

Binarización	
FUNCIÓN	DESCRIPCIÓN
filtradoBinario.m	Realiza un filtrado binario, útil únicamente para imágenes binarias.
FingerPrint_MedianFilter.m	Realiza un filtrado de media a nivel binario, útil únicamente para imágenes binarias.
Pol_Chebichev.m	Determina los polinomios de Chebichev.
BinDireccional.m	Determina la segunda derivada direccional de una imagen para cada píxel.

Deteccion de minucias	
FUNCIÓN	DESCRIPCIÓN
Num_Crossing.m	Calcula el número de cruces por cero para un píxel de la imagen esqueletizada.
Es_PuntoBifurcacion.m	Determina si un punto de una cresta adelgazada ya fue etiquetado como la posición de alguna minucia de bifurcación.
Es_PuntoFinalizacion.m	Determina si un punto de una cresta adelgazada ya fue etiquetado como la posición de alguna minucia de finalización.
Hallando_ProximoElemento.m	Determina la posición siguiente para el algoritmo de seguimiento de crestas.
Ir_ProximoElemento.m	Permite desplazarse al próximo elemento para el algoritmo de seguimiento de crestas.
PointMinusRamdon.m	Genera un conjunto de puntos aleatorios cuyas posiciones pertenezcan al primer plano de la imagen de regiones.
dibujar_branchpoint.m	Dibuja las minucias de bifurcación superpuesta a la imagen en escala de grises.
dibujar_bridgespoint.m	Dibuja los puentes superpuestos a la imagen en escala de grises.
dibujar_endpoint.m	Dibuja las minucias de finalización superpuesta a la imagen en escala de grises.
dibujar_minucias.m	Dibuja las minucias de bifurcación y finalización superpuesta a la imagen en escala de grises.
dibujar_RutasBranchpoint.m	Dibuja las rutas vinculadas a las minucias de bifurcación y sus direcciones superpuestas a la imagen en escala de grises.
dibujar_RutasEndpoint.m	Dibuja las rutas vinculadas a las minucias de finalización y sus direcciones superpuesta a la imagen en escala de grises.
dibujar_RutasEnd_Branch.m	Dibuja las rutas vinculadas a las minucias de bifurcación y finalización y sus direcciones superpuestas a la imagen en escala de grises.
mostrar_listas.m	Muestra las listas de las posiciones de las minucias en la línea de comandos de MATLAB.

6.6.3. Funciones para la extracción del Finger Code

6.6.3.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
FingerPrint_KernelFingerCode.m	Genera las máscaras del banco de filtros de Gabor bidimensional.
FingerPrint_Normalizacion.m	Normaliza los sectores circulares centrados en el punto de referencia de la impresión dactilar.
FingerPrint_ImagenesFiltradas.m	Genera las imágenes filtradas por el banco de filtros de Gabor bidimensional.
FingerPrint_Caracteristicas.m	Calcula el *Finger Code* vinculado a las imágenes filtradas.
FingerPrint_RotCaracteristicas.m	Genera los vectores de características rotados y los almacena en un archivo de texto.
FingerPrint_RotCaract_Unit.m	Genera los vectores de características rotados pero no los almacena en un archivo de texto.
FingerPrint_RotCaractMatching.m	Genera dos vectores de características rotados y los almacena en un archivo de texto, el primero vinculado a los sectores circulares sin desplazar y el segundo vinculado a los sectores circulares desplazados 11.5º
FingerPrint_GeneracionBaseDatos.m	Almacena en un única estructura todos los *Finger Code* de la base de datos DB2 del FVC2002
prueba_UserGenFC_unit.m	Aplicación

6.7. Funciones para la coincidencia de impresiones dactilares

6.7.1. Funciones para la técnica basada en correlación cruzada normalizada

6.7.1.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
SingularCoh.m	Determina las singularidades (puntos máximos) de la matriz de coherencia que pertenecen al primer plano de la imagen
Fingerprint_SingularCohLQV.m	Agrupa en un determinado número de centroides las singularidades de la matriz de coherencia utilizando cuantización vectorial.
GenerarDatosCORRELACION.m	Determina las singularidades agrupadas de una imagen para la coincidencia por correlación cruzada normalizada.
GenerarTemplateCORRELACION.m	Almacena en un único vector todas singularidades agrupadas de una base de datos.
Fingerprint_MatchNorCorrLQV.m Fingerprint_MatchNorCorrLQVopt.m	Permite realizar la coincidencia de puntos característicos en imágenes a través del cálculo de la correlación cruzada normalizada.
Fingerprint_MatchingCORRELACION.m	Determina el índice de coincidencia producto de realizar la correlación cruzada normalizada de una imagen de prueba versus las imágenes almacenadas en la base de datos.
FaseConsolidacion.m	Determina la correspondencia entre PRP a través de un umbral elíptico.
prueba_CoincidenciaCORRELACION_Total.m	Aplicación de verificación.
prueba_TestCoincidenciaCORRELACION.m	Aplicación para la determinación de la FAR y FRR.

6.7.1.2. Funciones de Soporte

FUNCIÓN	DESCRIPCIÓN
AgrupandoDatos.m	Agrupa puntos vecinos a través de una técnica heurística.
local_sum.m	Determina la matriz de sumas acumuladas
SuavizamientoGradiente.m	Calcula el Vector Gradiente con máscaras extendidas.

6.7.2. Funciones para la técnica basada en el alineamiento de minucias a través de la triangulación de Delaunay

6.7.2.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
TripletsDelaunay.m	Determina las tripletas de puntos vinculadas a los triángulos de Delaunay correspondientes a un vector de minucias.
TripletsMinucias.m	Obtiene las invarianzas de Triangulación vinculado a una tripleta de puntos.
ObtenerTripletsMinucias.m	Obtiene las invarianzas de Triangulación vinculado a un vector de minucias.
MejorAlineamientoTRI.m	Determina las invarianzas de Triangulación coincidentes vinculadas a dos vectores de minucias.
ObtenerTransfRigida2.m	Calcula la transformación rígida vinculada a dos tripletas de puntos supuestamente vinculadas.
TransformSimilaridad.m	Determina la transformación rígida que genera la máxima coincidencia de minucias.
AplicandoTransfRigida.m	Aplica una transformación rígida a un vector de puntos minucia.
ValidarTransRigida.m	Valida la transformación rígida a través de la correlación de cruzada normalizada.
CuentaCoincidencias.m	Cuenta el número de minucias coincidentes sobre dos vectores de minucias de puntos.
GenerarDatosTRIPLETS.m	Determina el vector de características para la coincidencia por el alineamiento de minucias.
GenerarTemplateTRIPLETS.m	Almacena en un único vector todas las minucias, tripletas de puntos y invarianzas de triangulación de una base de datos.
Fingerprint_MatchingTRIPLETS.m	Determina el índice de coincidencia producto de realizar el alineamiento de los vectores minucias de una imagen de prueba versus las minucias de las imágenes almacenadas en la base de datos a través de la triangulación de Delaunay.
prueba_CoincidenciaTRIPLETS_Total.m	Aplicación de verificación
Prueba_GeneracionTRIPLETS_Total.m	Aplicación para la determinación de las tripletas de minucias vinculadas a todas las imágenes de la base de datos DB2 del FVC.
prueba_TestCoincidenciaTRIPLETS.m	Aplicación para la determinación de la FAR y FRR.

6.7.2.2. Funciones de Soporte

Generacion de Puntos	
FUNCIÓN	DESCRIPCIÓN
datos.m	Genera un vector de minucias.
DatosPerturbados.m	Aumenta ruido a las posiciones de las minucias generadas.
DatosPolar.m	Determina la representación polar de las posiciones de las minucias.
EliminacionPuntos.m	Elimina minucias en forma aleatoria del vector de minucias generado.
RotDespl_Puntos.m	Rota y desplaza las posiciones de las minucias.
DibujarDatosPolar.m	Dibuja las minucias en un sistema coordenado polar.
rotar_escalar.m	Rota y escala las posiciones de las minucias generadas.
PerturbacionMinucias.m	Aumenta ruido a las orientaciones de las minucias generadas
prueba2.m	Aplicación de las funciones de generación de minucias y variación en sus posiciones y orientaciones.

Coincidencia de minucias	
FUNCIÓN	DESCRIPCIÓN
dibujar_PointsMinus.m	Dibuja las posiciones de las minucias.
dibujar_vminucias.m	Dibuja las posiciones y las orientaciones de las minucias.
dibujo_PuntosCoincidentes.m	Dibuja puntos coincidentes de dos vectores de minucias.

6.7.3. Funciones para la técnica basada en el FingerCode

6.7.3.1. Funciones Principales

FUNCIÓN	DESCRIPCIÓN
DistanciaEuclidea.m	Compara dos *Finger Code* a través de la distancia Euclidiana.
Fingerprint_MatchingFingerCode.m	Determina el índice de coincidencia producto de realizar la comparación del *Finger Code* de una imagen de prueba versus los *Finger Code* de las imágenes almacenadas en la base de datos a través del cálculo de la distancia Euclidiana.
GenerarTemplateFingerCode.m	Almacena en una única estructura todos los *Finger Code* de una base de datos de plantillas.
prueba_CoincidenciaFINGERCODE_Total.m	Aplicación de verificación.
prueba_TestCoincidenciaFINGERCODE.m	Aplicación para la determinación de la FAR y FRR.

6.8. Funciones de Soporte General

Lectura y visualización de imágenes	
FUNCIÓN	DESCRIPCIÓN
obtener_imagen.m	Obtiene la imagen en escala de grises vinculado a un archivo de imagen de extensión tif y bmp.
ObtenerNombreArchivo.m	Devuelve una cadena de texto que contiene el nombre de un archivo sin tomar en cuenta la extensión.
DibujarImagen.m	Dibuja una imagen binaria o en escala de grises.
DibujarSeccionImagen.m	Permite dibujar una imagen y resaltar una sección determinada.
DibujarPuntos.m	Dibuja puntos específicos sobre una imagen binaria o en escala de grises.

Funciones de dibujo	
FUNCIÓN	DESCRIPCIÓN
cuadrados.m	Dibuja un cuadrado con una orientación específica visto desde su centro.
cuadrado_org.m	Dibuja un cuadrado con una orientación específica visto desde la esquina superior izquierda.
triangulo.m	Dibuja un triángulo de coordenadas específicas visto desde su centro.
circle.m	Dibuja un círculo visto desde su centro.
elipse.m	Dibuja una elipse visto desde su centro.
drawseg.m	Dibuja una serie de segmentos de línea.
vectores.m	Dibuja un vector.
GenPlanoSen.m	Dibuja una superficie sinusoidal a una frecuencia, orientación, fase y nivel de dc específica

Operaciones sobre matrices	
FUNCIÓN	DESCRIPCIÓN
redimensionar_imagen.m	Redimensiona una matriz para poder sectorizarla en bloques de tamaño específico.
Extraccion.m	Extrae una vecindad centrada de una matriz considerando el efecto de la extracción de vecindades en los bordes de una matriz.
ExtraccionRot.m	Extrae una vecindad centrada de una matriz con una orientación específica considerando el efecto de la extracción de vecindades en los bordes de una matriz.
ExtraccionSinRelleno.m	Extrae una vecindad centrada de una matriz sin considerar el efecto de la extracción de vecindades en los bordes de una matriz.
GrabarMatrizImagen.m	Normaliza una matriz y la almacena como un archivo de imagen con extensión tif.
umbral_dinamic.m	Determina el valor umbral vinculado a una matriz.

Visualización de datos de una matriz	
FUNCIÓN	DESCRIPCIÓN
DibujarMatriz_2.m	Dibuja una matriz como si fuera una imagen.
DibujarSuperficieMatriz.m	Dibuja una matriz como una superficie y como imagen.
DibujarContornoMatriz.m	Dibujar las curvas de nivel vinculadas a una matriz.
plotmat.m	Dibuja una matriz con caracteristicas específicas sobre el formato del texto.
matprint.m	Muestra una matriz en la línea de comandos de MATLAB.
plotPoint.m	Dibuja puntos especificando el tipo de marca y el texto asociado.

Funciones matematicas	
FUNCIÓN	DESCRIPCIÓN
angle_estimation	Estima el ángulo de un par coordenado considerando el cuadrante.
angle_estimation_center.m	Estima el ángulo normal de un par coordenado considerando el cuadrante.
angle_nor.m	Obtiene el ángulo normal de un ángulo cualquiera.
divisores.m	Devuelve un vector con todos los posibles divisores de cualquier número entero.

Funciones variadas	
FUNCIÓN	DESCRIPCIÓN
noise.m	Añade ruido a una imagen.
CentroMasas.m	Determina el centro de masas de una imagen binaria.
IndiceCalidad.m	Determina el índice de calidad de una imagen.
confmat.m	Calcula la matriz de confusión.
conffig.m	Muestra la matriz de confusión.
ValidandoSingularidad.m	Elimina aquellas singularidades cercanas a los bordes.
PuntoRotado.m	Rota y traslada un punto coordenado bidimensional.

6.8.1. Funciones para el filtrado de imágenes

FUNCIÓN	DESCRIPCIÓN
putborde.m	Extiende los bordes de una imagen.
EliminarBordes.m	Establece a un valor específico los bordes de una matriz.
conv2b.m	Realiza la convolución bidimensional considerando el efecto de los bordes y devuelve una imagen del mismo tamaño que la de entrada.
Gabor_mask.m y mask_Gabor_2D	Genera la máscara de Gabor bidimensional.
mask_Gaussiano_1D.m	Determina la máscara Gaussiana unidimensional.
mask_Gaussiano_2D.m	Determina la máscara Gaussiana bidimensional.

6.8.2. Funciones para el cálculo del Vector Gradiente de una imagen

FUNCIÓN	DESCRIPCIÓN
FingerPrint_CalculoGradiente.m	Determina el Vector Gradiente de una imagen.
MatrizHessiana.m	Determina la Matriz Hessiana de una imagen.
TensorGradiente.m	Determina el Tesor Gradiente de una imagen.

FUNCIÓN	DESCRIPCIÓN
MaximosMinimosImagen.m	Determina los máximos y mínimos de una imagen.
filtrado_binario.m	Realiza un filtrado binario.

6.9. Resumen

En este capítulo describimos la biblioteca de funciones o *toolbox* desarrollada en este trabajo, la cual denominamos *"Toolbox* para el Reconocimiento por Huellas Dactilares – TReHD", describimos las ventajas de haberla implementado en MATLAB frente a otras alternativas y su utilidad como herramienta fundamental para futuras investigaciones e implementaciones tales como las descritas en el siguiente capítulo. Ademas, describimos brevemente cada una de las funciones conformantes del *toolbox*, cubriendo las fases de Análisis, Extracción de

caracteristicas y Coincidencia de impresiones dactilares, cuyos algoritmos fueron explicados en detalle en los capítulos III, IV y V respectivamente, incluyendose tambien las funciones base y de soporte general, que permiten la visualización de datos y caracteristicas de las imagenes, entre otros usos.

Capítulo VII
Verificación de personas por Huellas Dactilares basada en DSP

7.1. Introducción

Con el gran avance en el desarrollo de los sistemas embebidos (*embedded systems*), la verificación por huellas dactilares viene migrando rápidamente hacia el mundo de los equipos móviles, convirtiéndose en una parte crítica para aplicaciones tales como los sistemas de acceso, PDA's, *notebooks* e incluso teléfonos celulares. Por lo tanto, la verificación de la identidad juega un rol básico debido a que permite el acceso a lugares estratégicos y recursos a ser controlados.

El problema de la verificación de personas por huellas dactilares en tiempo real se puede tratar bastante bien utilizando como plataforma de hardware un computador de escritorio. Sin embargo, cuando se implementa esta aplicación en un dispositivo embebido encontramos ciertas dificultades causadas por la menor velocidad de la CPU, por la ausencia de una memoria caché e incluso en ciertas situaciones por la ausencia de una unidad de punto flotante. La optimización del proceso de verificación no implica sacrificar la confiabilidad, por lo tanto se requiere una técnica rápida, de bajo costo y precisa para desarrollar aplicaciones embebidas.

Debido a que los algoritmos de extracción de características y coincidencia involucran una gran cantidad de cálculo, no es una tarea trivial realizar éstos en dispositivos embebidos. Existen varias técnicas que pueden ser utilizadas para la verificación de personas por huellas dactilares en sistemas embebidos. Éstas incluyen los circuitos integrados de aplicación específica (*Application Specific Integrated Circuit* – ASIC), los procesadores digitales de señal (*Digital Signal Processor* – DSP), los sistemas de sólo un chip (*System-on-Chip* - SoC), entre otros.

En este capítulo se describe nuestra implementación algorítmica de un verificador de personas por huellas dactilares en una plataforma de desarrollo DSP basada en el procesador digital de señales TMS320C6711 de *Texas Instruments* que permite autenticar la identidad declarada por un individuo, comparando las características extraídas de su impresión dactilar con la registrada previamente.

Tal como fue presentado en el capítulo II, la tarea de reconocimiento en el modo de verificación requiere que una persona declare su identidad, por ejemplo a través de un código personal o PIN, luego el sistema compara directamente sus características biométricas con la característica asociada al código personal [Jain et al., 2004]. El resultado es una decisión binaria "si" o "no", que permite o deniega el acceso al sistema.

El diagrama general del proceso de verificación por huellas dactilares se muestra en la sección 2.2.2., su funcionamiento puede resumirse en los siguientes pasos:

1. El usuario ingresa su identidad (mediante un teclado, tarjeta magnética, entre otros), así como su característica biométrica requerida (en este caso, su huella dactilar). Obsérvese que en las aplicaciones de identificación el usuario no suministra su identidad.
2. El sistema extrae la representación de la característica biométrica de entrada, que consiste en la obtención de la respectiva plantilla (usualmente unos centenares de bytes).
3. Se compara la plantilla de entrada con la plantilla almacenada vinculada a la identidad suministrada previamente. Esta representación (la plantilla almacenada) se obtiene durante la inscripción de dicha persona.
4. El resultado de la comparación es una distancia o probabilidad de semejanza entre ambas plantillas. Se compara esta distancia con un valor umbral, si es menor que el valor umbral se acepta al usuario; en caso contrario se niega el acceso. Uno de los aspectos claves es la determinación de dicho valor umbral, puesto que condiciona los valores de FMR y FNMR descritos en la sección 2.5., en algunas aplicaciones interesa que una de

las dos tasas sea baja a costa de elevar la otra.

7.2. Herramientas de desarrollo utilizadas

El diseño de un sistema de verificación requiere las consideraciones de costo, tiempo de desarrollo y desempeño en tiempo real. Debido a la gran cantidad de datos que deben ser procesados y además a la complejidad computacional de los algoritmos empleados, todo sistema de verificación requiere una plataforma de desarrollo que provea herramientas de soporte tanto de hardware como de software. De acuerdo con el IBG, actualmente *Texas Instruments* ofrece una de las plataformas de desarrollo más flexibles de la industria biométrica, pues ha añadido a sus Plataformas de Desarrollo DSP (*DSP Starter Kit* - DSK) la posibilidad de conectarse a un dispositivo externo que realiza la captura de las huellas dactilares denominado Herramienta de Desarrollo para Verificación por Huellas Dactilares (*Fingerprint Authentication Development Tool* - FADT) a través de un bus de expansión, suministrando así una efectiva herramienta de bajo costo para construir sistemas de verificación por huellas dactilares con los productos basados en los procesadores de *Texas Instruments*, tal como se muestra en la figura 7.1.

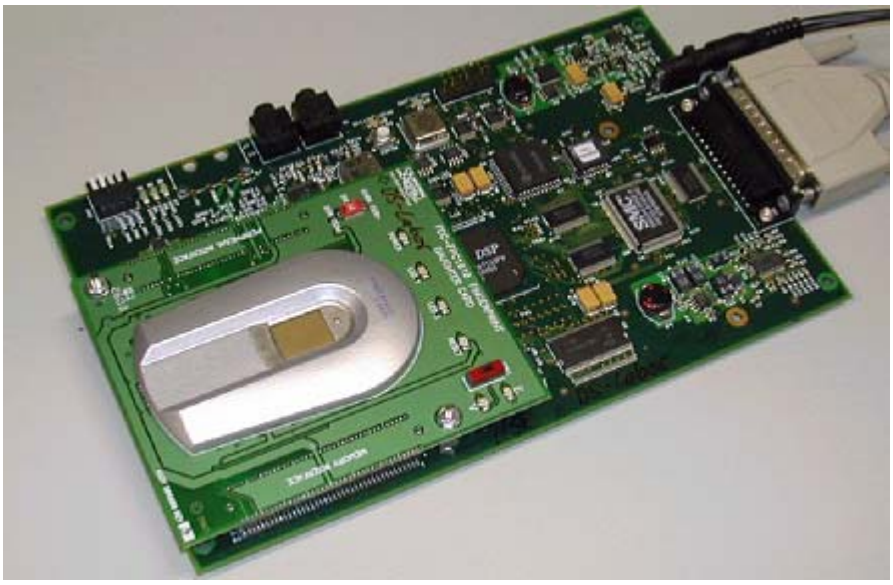


Figura 7.1 Tarjeta de desarrollo DSP TMS320C6711 conectada a la FADT. Imagen obtenida de [Plaz y Lang, 2004].

Utilizando las herramientas de hardware y software incluidas en el DSK es posible la construcción de un sistema autónomo de verificación por huellas dactilares, representado a través del diagrama de bloques de la figura 7.2. El hardware involucrado consiste básicamente en:

(a) Un bloque de captura de la huella dactilar a través del sensor capacitivo FPC1010 que es monolítico versátil y compacto (ver sección A.3.2.2.).

(b) Un bloque de extracción de características y coincidencia realizado por el procesador DSP TMS320C6711 junto con las memorias externas incluidas en el DSK.

(c) Una interfaz serial se incluye también en el DSK, que permite al DSP comunicarse con cualquier dispositivo de control y para conectarse al computador durante la inscripción de un nuevo usuario.

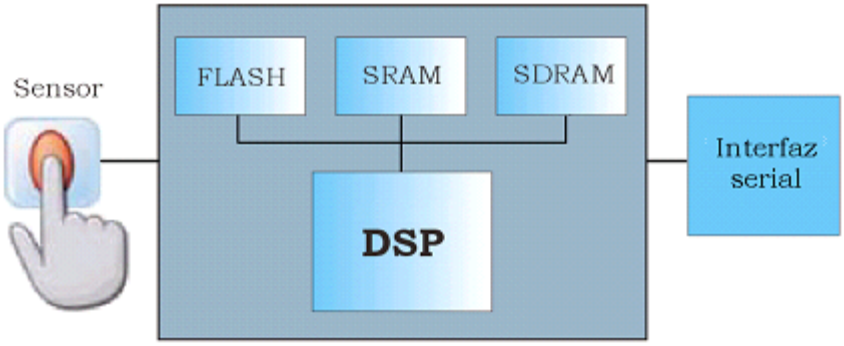


Figura 7.2 Diagrama de bloques de la plataforma de verificación provista por las herramientas de TI.

A pesar de que las herramientas descritas en el párrafo anterior estuvieron fuera de nuestro alcance, gracias al Laboratorio de la Facultad de Ingeniería Electrónica – UNMSM utilizamos el DSK TMS320C6711 de dicho laboratorio con el objetivo de probar la funcionalidad de la técnica de verificación seleccionada y para compensar la ausencia del dispositivo de captura, se utilizaron las bases de datos estándares para la evaluación de los sistemas de verificación, descritas en la sección 5.5.1. El anexo E presenta una descripción detallada de las principales características del hardware y software que provee el DSK TMS320C6711, las cuales

han sido empleadas para el desarrollo del verificador, así como también para realizar las pruebas experimentales.

7.3. Descripción del procedimento desarrollado

El verificador diseñado opera de la siguiente forma:

La fase *Off-Line* (Inscripción): Se capturan las impresiones dactilares de un conjunto de individuos a inscribirse, realizando la extracción de características y almacenando éstas como plantillas en una base de datos o en la memoria del DSP, para su uso posterior.

La fase *On-Line* (Verificación): El individuo a ser verificado proporciona su identidad y luego su impresión dactilar es capturada a través del sensor biométrico, seguidamente el procesador DSP recibe la imagen extrayendo las características o plantilla de entrada, siendo ésta comparada con la plantilla vinculada a la identidad proporcionada la cual ha sido previamente almacenada.

Puesto que las impresiones dactilares empleadas en nuestros experimentos están almacenadas en una base de datos, el proceso de captura descrito anteriormente se realiza seleccionando la imagen correspondiente.

Por lo tanto, el DSP debe realizar las siguientes tareas:

1. Recibir y almacenar en memoria la(s) plantilla(s) transferida(s) por la aplicación en el computador durante la inscripción de los usuarios.
2. Recibir y procesar la imagen de entrada transferida por el computador, tomando una decisión acerca de la autenticidad de la biométrica de entrada y finalmente enviar la respuesta al computador para su visualización.

Además, el computador debe realizar las siguientes tareas:

1. Permitir al usuario inscribirse en el sistema, utilizando una interfaz gráfica de usuario sencilla y amigable.
2. Durante la inscripción, establecer la comunicación con el DSP para enviarle la(s) plantilla(s) requerida(s).
3. Durante la verificación, enviar la imagen al DSP y esperar la respuesta

para mostrar una ventana de aceptación o rechazo al usuario que solicita el acceso.

7.3.1. Implementación en el DSP

En esta sección se describe la implementación de los algoritmos en el procesador TMS320C6711 para la tarea de verificación de personas por huellas dactilares, utilizando imágenes en escala de grises de ocho bits de profundidad, las que pueden ser obtenidas de cualquier tipo de escáner electrónico de huellas dactilares así como también de imágenes almacenadas en disco.

Se ha elegido la técnica de verificación basada en el análisis de texturas orientadas debido a la menor complejidad y al costo computacional que requiere su implementación frente a las técnicas de correlación y minucias que hemos desarrollado, como se indicó en la figura 5.15, en donde se muestra el buen compromiso que existe entre la complejidad algorítmica y el tiempo de procesamiento de las etapas de extracción de características y coincidencia. Por otra parte, tal como es remarcado en [Prabhakar, 2001, pp. 222 - 226] esta técnica de verificación resulta ser muy adecuada para implementarse en un sistema embebido tal como un procesador DSP de alto rendimiento.

Existen dos formas de implementar el código para el verificador de personas utilizando una plataforma DSP:

1. Codificar manualmente todos los algoritmos en lenguaje *assembler*. Esta técnica provee un alto desempeño, pero requiere un elevado tiempo de desarrollo.
2. Codificar inicialmente todos los algoritmos en el lenguaje estándar ANSI C para el DSP, convertirlo a lenguaje máquina usando el compilador del CCS y luego incluir progresivamente técnicas de optimización para reducir el tiempo de procesamiento [Texas Instruments, 2001e].

Decidimos utilizar la segunda forma pues permite reducir el tiempo de desarrollo, luego empleamos el lenguaje *assembler* sólo en las partes más críticas del código.

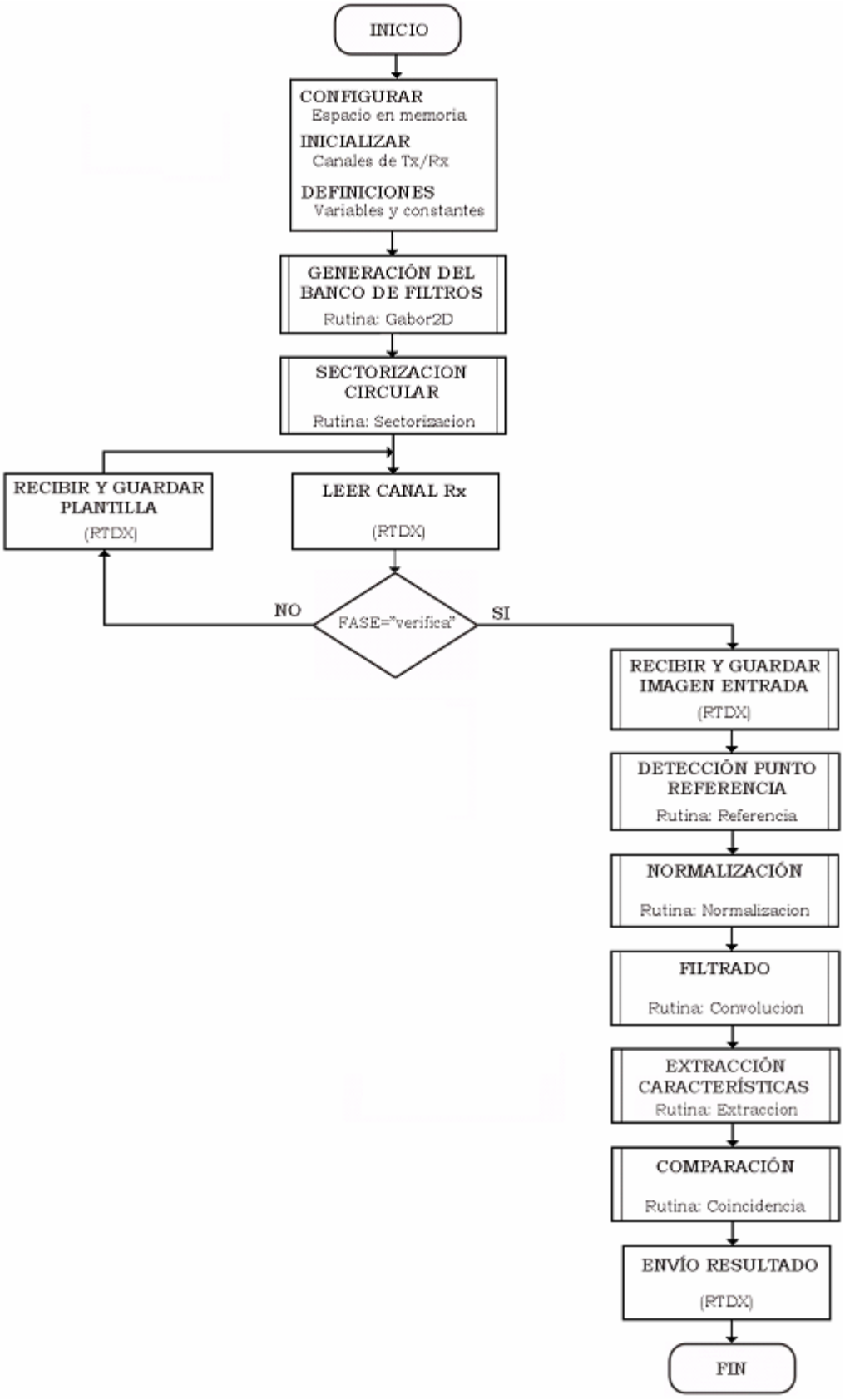


Figura 7.3 Diagrama de bloques de la rutina principal ejecutada por el procesador DSP.

La figura 7.3 muestra el diagrama de bloques de la rutina principal ejecutada desde el procesador DSP. El flujo se divide en dos partes, dependiendo del estado de la variable "fase" que indica al procesador si debe almacenar la plantilla en memoria o si debe realizar la verificación de la imagen de entrada. Los bloques que contienen (RTDX) realizan la comunicación en tiempo real con el computador utilizando uno de los canales de transmisión o recepción inicializados en el principio de la rutina a través de la librería de funciones que provee el CCS ubicada en el archivo de cabecera "rtdx.h", que permite establecer el enlace entre el DSP y el computador.

7.3.1.1. Rutina Gabor2D

Esta rutina realiza la generación del banco de filtros de Gabor bidimensional (ver figura 7.4), es decir crea las máscaras de Gabor en las ocho orientaciones predefinidas, tal como fueron establecidas en la sección 4.4.1.4., las cuales son utilizadas posteriormente para realizar el filtrado en el dominio espacial. Debido a las propiedades de la función de Gabor bidimensional, la creación de las máscaras requiere de las funciones matemáticas provistas por las librerías del estándar ANSI C, localizadas en el archivo de cabecera "math.h". Además, puesto que estas máscaras no cambian en el tiempo, no es necesario volver a calcularlas cada vez que se presenta una nueva imagen de entrada, por lo tanto el tiempo de procesamiento que requiere esta rutina no es considerado como parte del tiempo de extracción de características, sino más bien como parte del tiempo de inicialización del procesador.

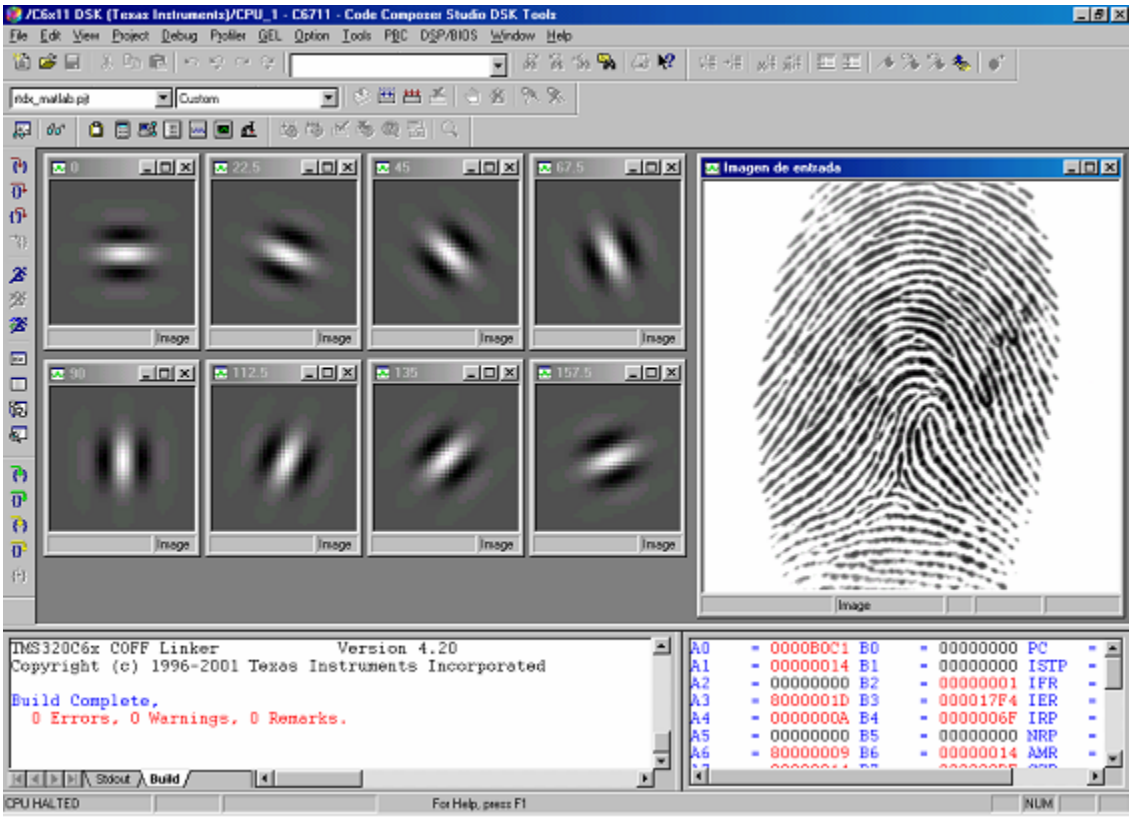


Figura 7.4 Implementación del banco de filtros de Gabor 2D a través del CCS. Imagen 1_1.tif de la base de datos 1 del FVC.

7.3.1.2. Rutina Sectorizacion

Esta rutina realiza la sectorización circular de la región de interés proveniente de la rutina Referencia (ver figura 7.5), tal como se describió en la sección 4.4.1.2., la cual consiste básicamente en crear una matriz del mismo tamaño que la imagen recortada, que contenga las etiquetas del sector al que pertenece el píxel correspondiente de la región de interés. Debido a las propiedades geométricas de la sectorización, se requiere utilizar las funciones matemáticas del estándar ANSI C, localizadas en el archivo de cabecera "math.h". Puesto que la sectorización circular es la misma para cualquier imagen de entrada, no es necesario calcular nuevamente las etiquetas para cada nueva imagen, por lo tanto el tiempo de procesamiento que requiere puede ser considerado como parte del tiempo de inicialización.

7.3.1.3. Rutina Referencia

Esta rutina realiza la extracción del punto de referencia para determinar la región de interés (ver figura 7.5), tal como se describió en la sección 4.4.1.1. Ésta tiene como entrada a la imagen a ser verificada y genera una imagen recortada que tiene como centro el punto de referencia, donde se encuentra ubicada la región de interés que será procesada y comparada por el sistema. Esta rutina requiere realizar el cálculo de los Vectores Gradiente, promediándose luego a nivel de bloques y obteniendo el punto de referencia a través del procedimiento descrito en la sección 4.2.1. El tiempo que requiere este procesamiento si es considerado como parte del tiempo de extracción de características.

7.3.1.4. Rutina Normalizacion

Esta rutina realiza la normalización de cada uno de los sectores circulares (ver figura 7.5), tal como se describió en la sección 4.4.1.3. Ésta tiene como entradas a la imagen recortada y la matriz de etiquetas obtenidas de la rutina Sectorizacion y generado una nueva imagen normalizada sólo dentro de la región circular de interés. El tiempo que requiere esta rutina es parte del tiempo de extracción de características.

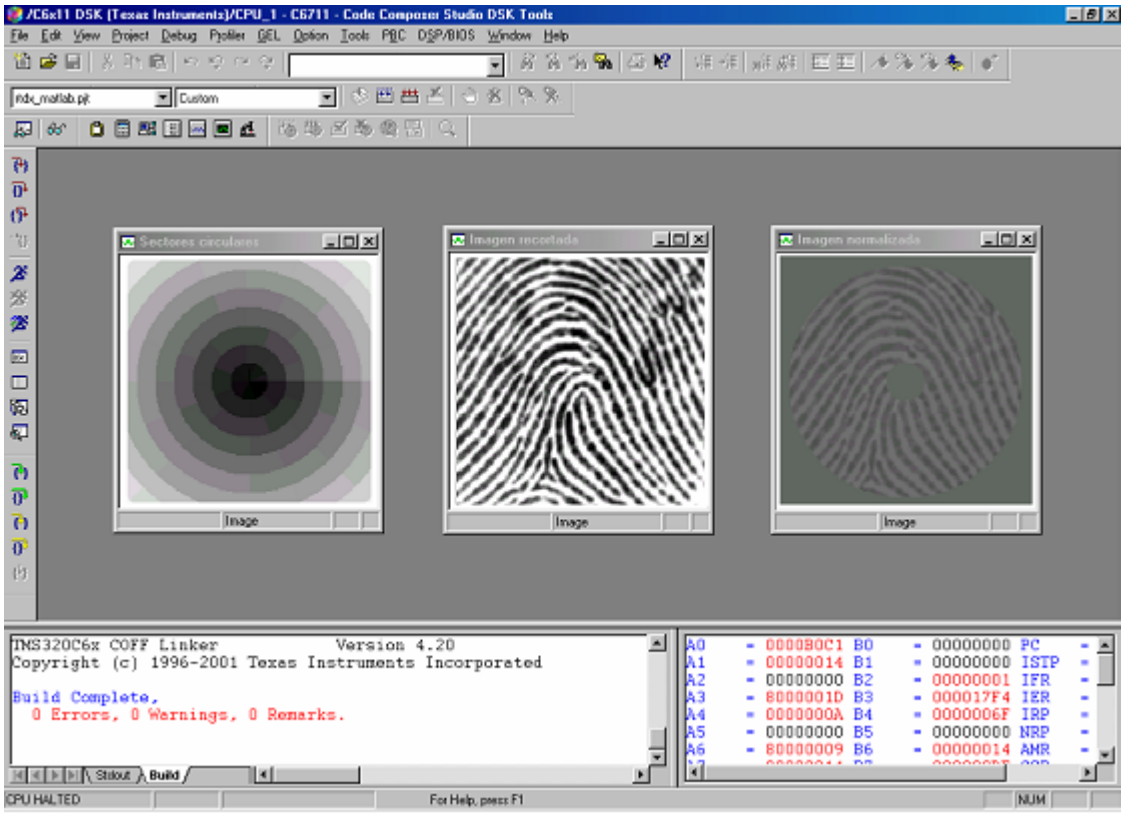


Figura 7.5 Implementación de los sectores circulares, detección del punto de referencia y la normalización de la región de interés a través del CCS. Imagen 1_1.tif de la base de datos 1 del FVC2002.

7.3.1.5. Rutina Convolucion

Esta rutina realiza el filtrado en el dominio espacial de la región de interés con las máscaras de Gabor obtenidas de la rutina Gabor2D (ver figura 7.6), que tiene como entradas las máscaras de Gabor y la imagen normalizada, generando las imágenes filtradas que a su vez son escaladas para que varíen en el rango de 0 a 255 debido a que se obtienen valores de píxeles grandes por el tamaño de las máscaras de convolución. Esta rutina representa cerca del 99% del tiempo de extracción de características, puesto que no sólo se requiere de un enorme número de operaciones de multiplicaciones y adiciones debido al tamaño de la imagen y de la máscara, sino también estas operaciones a su vez requieren del acceso de una gran cantidad de datos de la memoria (SDRAM externa), en donde se hayan ubicados los datos de entrada y de salida. El escalamiento es necesario para mantener

los valores dentro del rango de trabajo de las variables en el procesamiento posterior.

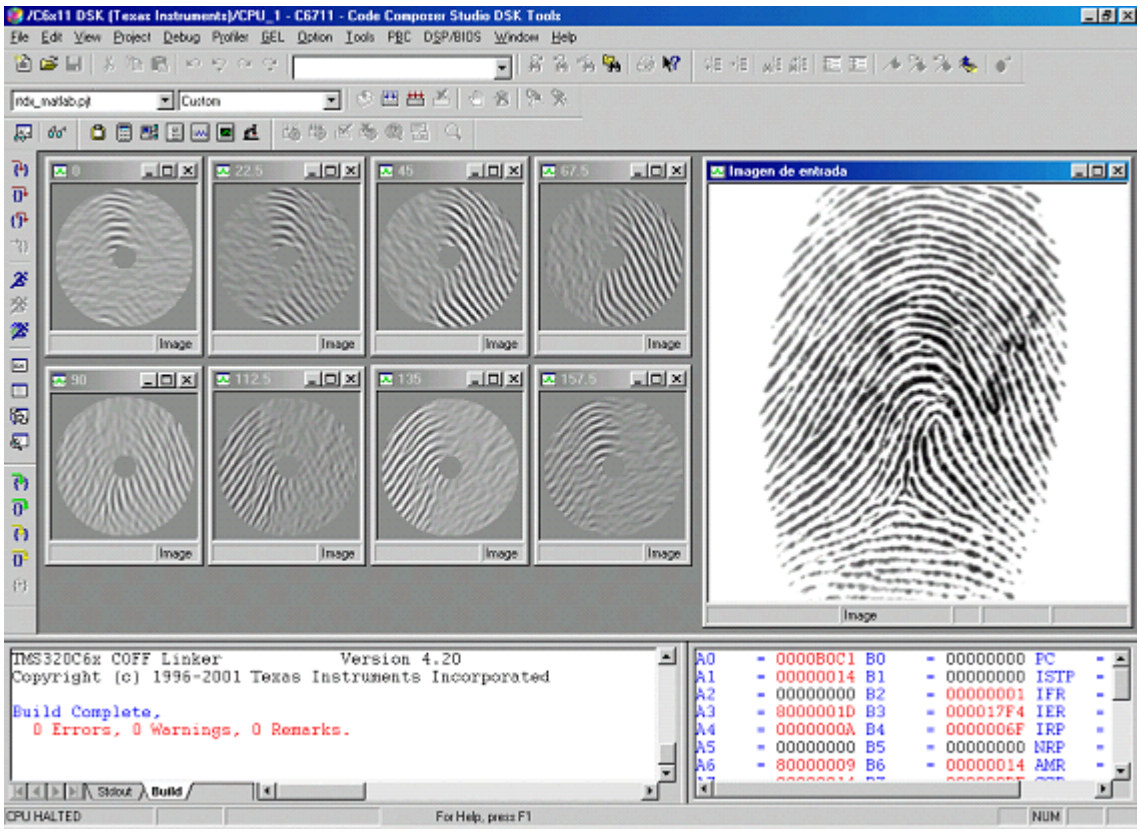


Figura 7.6 Implementación del filtrado de la región de interés a través del CCS. Imagen 1_1.tif de la base de datos 1 del FVC2002.

7.3.1.6. Rutina Extraccion

Esta rutina calcula el vector característico denominado *Finger Code* (ver figura 7.7) a través de la ecuación 4.53 que representa la varianza de la respuesta de los filtros en la imagen, tal como se describe en la sección 4.4.1.5. Ésta utiliza como entrada las imágenes filtradas y escaladas, generando un vector de características que contiene 640 valores (bytes) para cada imagen capturada.

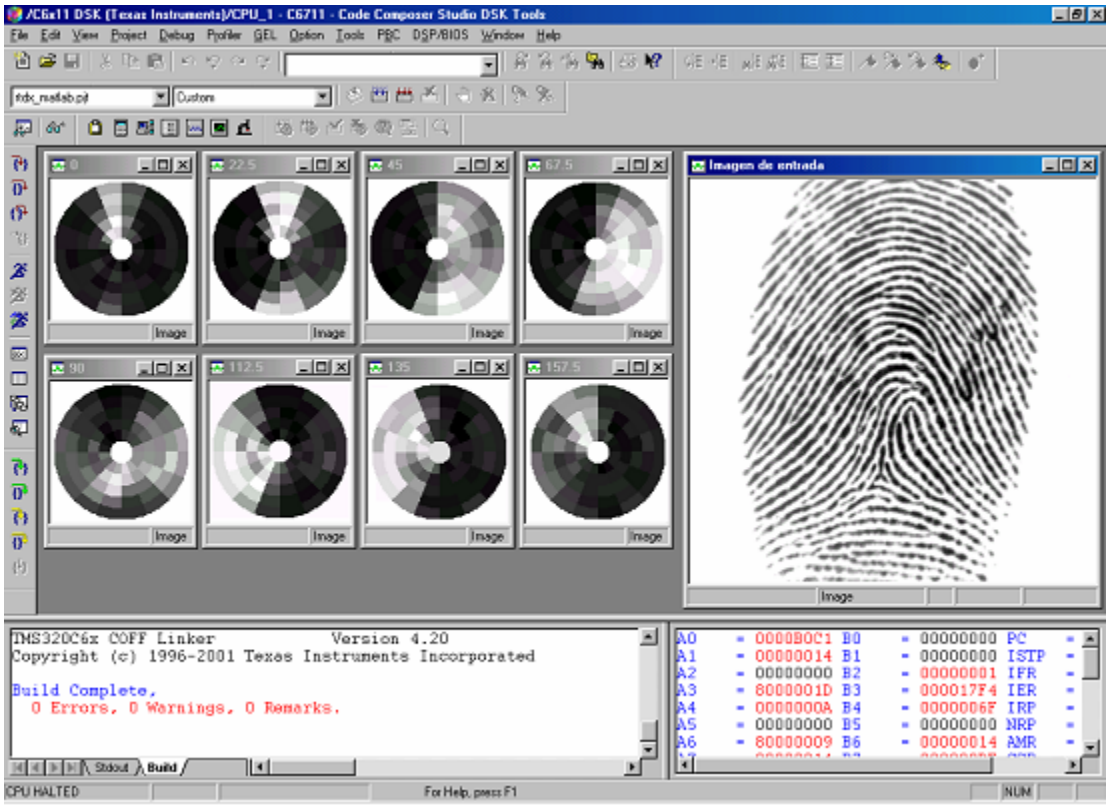


Figura 7.7 Implementación del extractor del vector de características a través del CCS. Imagen 1_1.tif de la base de datos 1 del FVC2002.

7.3.1.7. Rutina Coincidencia

Esta rutina realiza la comparación entre el vector de características extraído de la imagen de entrada con las versiones rotadas de las plantilla(s) previamente almacenada(s) en la memoria del DSP, tal como se describió en la sección 5.4.1. El resultado de esta rutina es un valor binario basado en un valor umbral que determina si la característica biométrica del usuario es auténtica o no.

7.3.2. Interfaz gráfica de usuario

La aplicación de comunicación entre el computador y el DSP fue realizada utilizando el software MATLAB [Pérez, 2002; pp. 1-42], en donde se diseñó la GUI mostrada en la figura 7.8 y además se programaron las rutinas de comunicación con el procesador DSP a través del RTDX (ver anexo E). Esta GUI posee dos funcionalidades: registrar y verificar.

La opción Registrar permite inscribir el nombre o PIN del usuario junto con su impresión dactilar previamente adquirida y almacenada en disco.
La opción Verificar permite autenticar la identidad del usuario comparando la impresión dactilar de entrada con la previamente registrada. El resultado es un mensaje de aceptación o rechazo de acceso.



Figura 7.8 GUI para las tareas de inscripción (registrado) y verificación. Imagen 1_1.tif de la base de datos 1 del FVC.

7.3.3. Resultados obtenidos

Un análisis detallado del desempeño de los algoritmos de verificación empleados fue descrito en el capítulo V. En esta sección se describen brevemente los tiempos de procesamiento de las principales rutinas que realiza el procesador DSP para la verificación.

Para acelerar el proceso de desarrollo de los algoritmos, se realizó la codificación de todas las rutinas empleando lenguaje estándar ANSI C (como se menciona en la sección 7.3.1.). Sin embargo, la extracción de características posee un elevado tiempo de procesamiento, principalmente la rutina de convolución espacial, debido al gran número de operaciones que realiza [Embree y Bimble, 1991; pp. 381-401], por lo tanto se utilizaron técnicas de optimización a éstas, asignando racionalmente los tipos de datos adecuados para aprovechar el multiplicador de 16 bits, añadiendo rutinas que operan en lenguaje *assembler* en las partes más criti-

cas del código, además de otras optimizaciones realizadas a nivel del compilador y descritas en [Texas Instruments, 2001f]. Como resultado se obtuvo una considerable reducción en el tiempo de procesamiento.

Adicionalmente a estas técnicas de optimización, es posible acelerar aún más el proceso de convolución si se realiza en el dominio de la frecuencia, lo que requiere transformar tanto las máscaras de Gabor como la imagen de entrada recortada. Sin embargo, los requerimientos de memoria se hacen mucho mayores, además resulta poco útil para una aplicación real implementar un algoritmo FFT utilizando el lenguaje ANSI C, éste debe estar optimizado tanto en memoria como en tiempo de procesamiento a nivel de *assembler*.

La figura 7.9 muestra una comparación entre los tiempos de extracción utilizando un computador Pentium MMX de 166 MHz con 128 MB de RAM frente al DSP de 150 MHz con 16 MB de RAM (antes y después de optimizarlo).

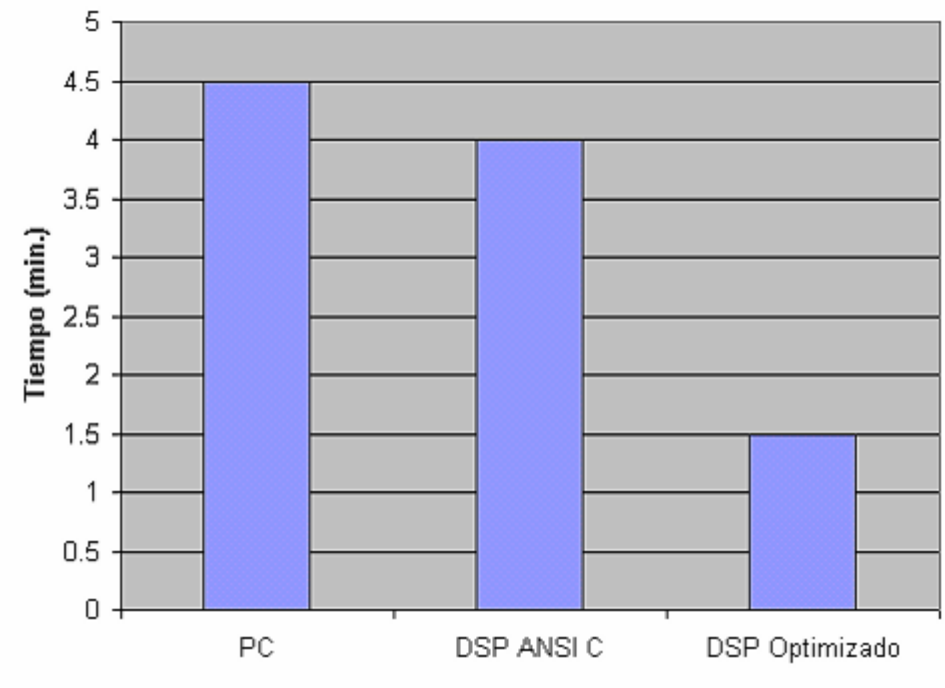


Figura 7.9 Comparación entre el tiempo de extracción de características en una PC, en el DSP y optimizado.

Es posible realizar la implementación utilizando únicamente rutinas en lenguaje *assembler* y *assembler* lineal, que permiten aprovechar el *pipeline*, reducir las

dependencias secuenciales del código para que puedan ejecutarse en paralelo y utilizar el DMA para acelerar los procesos de carga/almacenamiento en la memoria SDRAM, sin embargo este nivel de optimización requiere un tiempo de desarrollo exponencial respecto al presentado en este capítulo y está fuera de los objetivos planteados para este trabajo, pero debe ser considerado cuando se requiera la implementación de un producto comercial.

Es importante también resaltar una situación que debe ser considerada, al codificar un algoritmo en un procesador DSP de punto flotante se generan muchas más instrucciones en lenguaje máquina que en un procesador de punto fijo, tal como se demostró en [Chen et al., 2002], en donde se realizó una comparación entre el tiempo de procesamiento para un mismo algoritmo utilizando un procesador de punto flotante y uno de punto fijo, determinándose una proporción de uno a trece. Estos resultados parecen convertir a los procesadores de punto fijo como un candidato bastante fuerte para realizar el proceso de verificación, pues podría alcanzar tiempos de procesamiento inferiores a los tres segundos, lo que lo haría adecuado sobretodo en aplicaciones de tiempo real.

7.4. Resumen

Se ha implementado una técnica de verificación de personas en un DSP utilizando la técnica de representación basada en un banco de filtros, permitiendo aprovechar tanto las características locales como globales en las huellas dactilares para mejorar la toma de decisiones en la verificación de la identidad.

La técnica de verificación empleada ha sido seleccionada por su alta velocidad en la etapa de coincidencia y su baja complejidad computacional respecto a las otras técnicas que hemos desarrollado, este sistema realiza un buen desempeño principalmente cuando los requerimientos de desempeño de la aplicación no requieran de un valor muy alto de FMR.

Hemos comprobado de manera sencilla el correcto funcionamiento en cada etapa de los algoritmos codificados en el DSP, además han sido comparados con

los resultados obtenidos en el computador gracias a las herramientas de visualización gráfica provistas por el IDE del CCS, tal como se muestra en las figuras 7.4 al 7.7. Es importante recalcar las ventajas que presenta la optimización del código y la inclusión de rutinas en lenguaje *assembler* en las partes más críticas de éste, que reducen considerablemente el número de los ciclos de reloj que requiere su ejecución.

Capítulo VIII
Conclusiones y Trabajos Futuros

8.1. Conclusiones

El reconocimiento automático de personas por huellas dactilares es un problema complejo del reconocimiento de patrones, pues diseñar algoritmos capaces de extraer las características relevantes y coincidirlas en forma robusta es un problema muy difícil. No es exagerado afirmar que la investigación sobre este tema ha tenido principalmente como objetivo imitar el desempeño de un experto humano. Sin embargo, quizás la intuición humana usada como base para el reconocimiento manual no sea la más apropiada para diseñar sistemas de reconocimiento automático. Por tanto es necesario explorar nuevas características y técnicas mucho más ingeniosas que permitan mejorar el proceso de automatización.

Hemos analizado, implementado y comparado dos de los algoritmos más sofisticados para la estimación de la Imagen de Orientación, ambos algoritmos poseen tanto ventajas así como también desventajas, sin embargo una adecuada elección depende fuertemente de la aplicación. Además se han implementado y propuesto mejoras en las dos técnicas planteadas en la literatura para la estimación de la Imagen de Regiones: a nivel de píxeles y a nivel de bloques. En la segmentación por píxeles, se ha propuesto un procedimiento para obtener un conjunto de entrenamiento adecuado utilizando como entrada tres características de los píxeles, siendo éstas condensadas a través de dos SOFM para obtener un subconjunto representativo óptimo que es utilizado para la clasificación de las regiones. En la segmentación por bloques, se selecciona un conjunto de entrenamiento de manera similar al desarrollado en la segmentación por píxeles, pero con dos características diferentes y calculadas a nivel de bloques utilizando un

clasificador k-NN para la segmentación.

Para la determinación de los puntos singulares se han presentado dos técnicas, la primera basada en el índice de Poincaré y la segunda basada en los componentes sinusoidales de la Imagen de Orientación, llegando a la conclusión de que frente a la calidad y a la rotación de la imagen la segunda técnica demuestra ser más robusta, permitiendo además determinar de manera directa el punto de referencia de una impresión de tipo A y puesto que el tiempo de procesamiento y la complejidad computacional son mucho menores en la segunda técnica respecto a la primera, por lo que se recomienda su uso cuando se realicen procesos de inscripción de usuarios en línea y su posible implementación en sistemas embebidos.

Para la extracción de las minucias hemos propuesto una técnica para la determinación de sus atributos: tipo, posición y dirección, siendo el aporte de mayor interés la creación de un algoritmo de seguimiento de crestas robusto basado en reglas heurísticas plasmadas en una tabla de búsqueda (apendice A) que permite la determinación de la dirección de una minucia y un paso de post procesamiento para la eliminación de las minucias falsas. La técnica propuesta ha sido validada a través del indicador de rendimiento denominado Índice de Virtud y tal como se indica en la tabla 4.2 es robusta y exacta.

Para la técnica de coincidencia por correlación, proponemos usar la correlación local empleando las singularidades locales cuantizadas como centros geométricos de cada vecindad a comparar. Esta propuesta tiene como desventaja el gran costo en tiempo de procesamiento tanto en la extracción de características como en la fase de coincidencia.

Para la técnica de coincidencia basada en las minucias, hemos propuesto un algoritmo de coincidencia basado en el alineamiento de las minucias a través de la triangulación de Delaunay. Tal como lo indican las figuras 5.17

y 5.18 es la mejor de las tres técnicas implementadas, siendo la más adecuada para aplicaciones de alta seguridad y sobre todo presenta un tiempo de procesamiento aceptable.

Se ha implementado una técnica de verificación de personas sobre un DSP utilizando la representación basada en un banco de filtros de Gabor bidimensional, puesto que la extracción de características y la coincidencia son más factibles de implementarse en hardware que los métodos tradicionales basados en minucias, caracterizados porque su desempeño depende fuertemente de la calidad de la impresión dactilar. Ademas esta técnica posee una alta velocidad en la etapa de coincidencia y una baja complejidad computacional respecto a las otras técnicas que hemos desarrollado.

Hemos comprobado de manera simple el funcionamiento correcto en cada etapa acerca de los algoritmos codificados en el DSP, además se han comparado con los resultados obtenidos en el computador gracias a las herramientas de visualización gráfica provistas por el IDE del CCS, tal como se muestra en las figuras 7.4 al 7.7. Es importante recalcar las ventajas que presenta la optimización del código y la inclusión de rutinas en lenguaje assembler en las partes más críticas de éste, que reducen considerablemente el número de ciclos de reloj que requiere su ejecución.

Un componente clave en esta implementación ha sido el RTDX (ver anexo E), pues nos ha permitido establecer la comunicación e intercambiar datos, incluso imágenes en tiempo real desde el computador al DSP y por lo tanto evaluar adecuadamente el funcionamiento de la verificación. Sin embargo, esta valiosa herramienta posee como principal limitación un "ancho de banda" angosto, pues ha sido diseñado principalmente para aplicaciones de control.

8.2. Trabajos futuros

Una alternativa de gran interés en la etapa de extracción de minucias es su

detección a partir de la imagen en escala de grises, donde podría utilizarse la información brindada por el nivel de irregularidad (singularidades locales) y un algoritmo de validación basado por ejemplo en redes neurales para establecer que singularidad es o no una minucia.

Es de vital importancia disminuir el tiempo de coincidencia para la técnica basada en la correlación, aplicando por ejemplo algoritmos de correlación especializados (correlacionar únicamente los píxeles del primer plano o realizar un diezmado de la imagen a correlacionar). Para el caso de la técnica de coincidencia basada en minucias se puede aprovechar la información que brindan las rutas de las crestas vinculadas a cada minucia para obtener un alineamiento más robusto.

Una meta a corto plazo en el desarrollo del TReHD que es la optimización de sus funciones, a través de la inclusión del código en lenguaje C utilizando para ello archivos mex, tal que permita la evaluación de las técnicas de coincidencia implementadas en grandes bases de datos. Una meta a largo plazo es incluir: (a) funciones para la reconstrucción de impresiones dactilares, basadas por ejemplo en los modelos AM-FM [Acton et al., 2001], en la teoría espacio-escala, entre otros; (b) funciones para la compresión de imágenes especializadas para impresiones dactilares como son la WSQ [Ramos, 1997]; (c) funciones para clasificación de impresiones dactilares basadas en las técnicas de redes neurales, estructurales, entre otros.

El siguiente paso en el desarrollo de un verificador de personas por huellas dactilares presentado en este trabajo consiste en el análisis de los algoritmos empleados en este trabajo utilizando un procesador DSP de punto fijo, lo que nos asegura que servirá para los requerimientos de tiempo real que son necesarios para todo dispositivo autónomo.

Una considerable mejora en el tiempo de extracción de características del *Finger Code* puede ser realizada a través de la convolución en el dominio de la frecuencia, una excelente alternativa puede ser utilizar la biblioteca de

procesamiento digital de señales para la familia C6000 escrita en lenguaje *assembler* denominada DSPLIB para realizar la transformada rápida de Fourier bidimensional a través de la unidimensional, y aprovechar tanto la parte real como la compleja, procesando dos filas o columnas de la imagen en una sola transformación.

Varias mejoras pueden realizarse tanto para el procesamiento como para el reconocimiento por huellas dactilares a través de las herramientas de desarrollo DSP en tiempo real. Por ejemplo, puede emplearse una Plataforma de desarrollo EVM basada en el procesador DSP TMS320C6201 (de punto fijo), que posee una interfase PCI para establecer una comunicación mucho más rápida con el computador, de tal forma que permita realizar la tarea de reconocimiento fusionando la técnica basada en minucias (en el microprocesador) y la basada en texturas (en el procesador DSP) ejecutándose ambas en paralelo y tomando una decisión más acertada basada en ambos índices de similitud, obteniéndose mejores resultados en un tiempo de procesamiento mucho menor.

Existe una novedosa técnica para la detección directa de minucias desde las imágenes en escala de grises basada en un algoritmo iterativo de seguimiento de crestas [Maio y Maltoni, 1997], cuya ventaja es su bajo costo computacional, sin embargo ésta depende fuertemente de la calidad de la imagen, a pesar de ello parece ser un buen candidato para su implementación en un dispositivo embebido, por lo tanto representa una interesante alternativa de implementación futura.

También es factible realizar una etapa de pre procesamiento para la mejora de la imagen de entrada, sin embargo la mayoría de las técnicas actuales consumen un alto costo computacional. Por lo que un interesante aporte puede ser el de desarrollar una técnica que sea adecuada para su implementación en dispositivos embebidos.

Anexo A
Captura de Huellas Dactilares

A.1. Introducción

Las huellas dactilares son capturadas a través de uno de dos modos de adquisición: *off-line* (en diferido) y *on-line* (en tiempo real). Actualmente, muchos AFIS diseñados tanto para aplicaciones civiles como criminales capturan en tiempo real la huella dactilar a través de escáneres biométricos electrónicos, cuya estructura general es mostrada en la figura A.1. El sensor capta la superficie del dedo, generando una imagen analógica que es convertida en digital a través de un conversor A/D y la interfaz es la responsable de la comunicación con los dispositivos externos. La parte más importante de un lector o escáner biométrico es el sensor, puesto que es el componente en donde se forma la imagen.

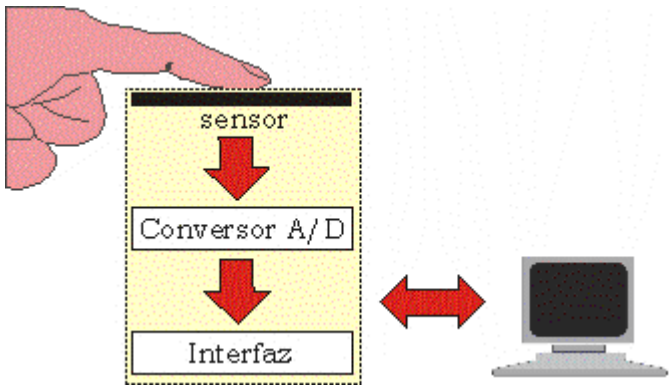


Figura A.1 Diagrama de bloques de un lector o escáner biométrico electrónico.

A continuación describiremos brevemente los principales parámetros técnicos que caracterizan a las impresiones dactilares y las técnicas empleadas para su adquisición.

A.2. Parámetros técnicos

Los principales parámetros técnicos que caracterizan a una impresión dactilar son:

Resolución: Indica el número de puntos o píxeles por unidad de longitud, usualmente especificado en pulgadas (*dots per inch* - dpi). La resolución mínima establecida para los sensores del FBI es 500 dpi. Probablemente la mínima resolución requerida para detectar con precisión las minucias esté en el intervalo de 250 a 300 dpi. Ejemplos de impresiones dactilares tomadas a diferentes resoluciones son mostradas en la figura A.2.

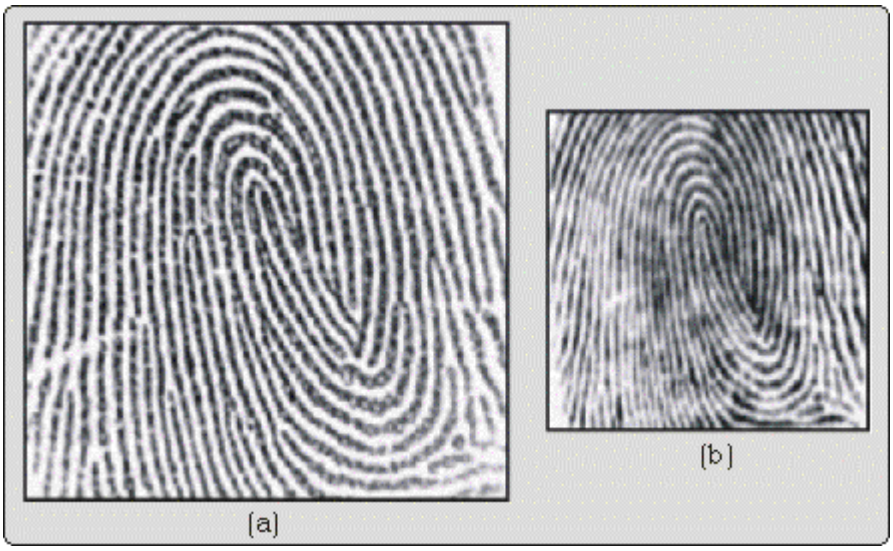


Figura A.2 Diferentes resoluciones de captura (a) 500 dpi; (b) 350 dpi.

Área de captura: Es el tamaño del área rectangular de captura. Una mayor área permite capturar más crestas y valles, proporcionando un patrón con mayor capacidad de discriminación. El área mínima requerida por las especificaciones del FBI es de 1x1 pulgadas cuadradas. Sin embargo, muchos sensores hoy en día poseen un área más pequeña, haciendo imposible que una huella dactilar completa sea capturada. Una pequeña área de captura mantiene tanto un costo bajo y un tamaño reducido, sin embargo también conlleva a innecesarios falsos rechazos.

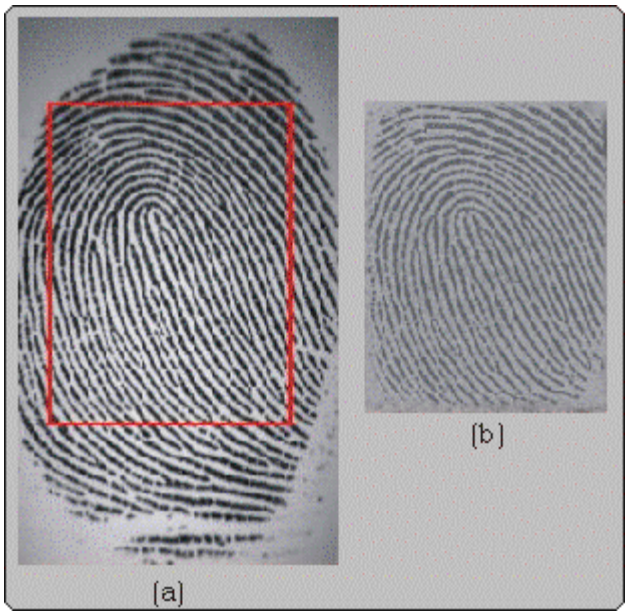


Figura A.3 Impresiones dactilares adquiridas por diferentes tipos de sensores (a) óptico; (b) capacitivo.

Rango dinámico (o profundidad): Es el número de bits utilizados para codificar el valor de la intensidad de cada píxel en la imagen. El estándar del FBI establece el uso de imágenes en escala de grises con una profundidad de ocho bits.

Precisión geométrica: Se define como la máxima distorsión geométrica presentada por el dispositivo de captura (ver figura A.4), se expresa como un porcentaje con respecto a las direcciones x e y. Muchos sensores ópticos presentan distorsiones geométricas que requieren ser compensadas pues alteran el patrón capturado.



Figura A.4 Distorsión geométrica presente en dos impresiones dactilares de la misma huella (a)101_2.tif; (b)101_3.tif. Imágenes de la base de datos 2 del FVC2002.

Calidad: Es un parámetro difícil de medir, especialmente debido a que es complicado separar la calidad de la imagen del estado intrínseco de la huella dactilar (ver figura A.5).

Todas las características anteriormente mencionadas trabajan juntas para establecer la precisión del sistema.

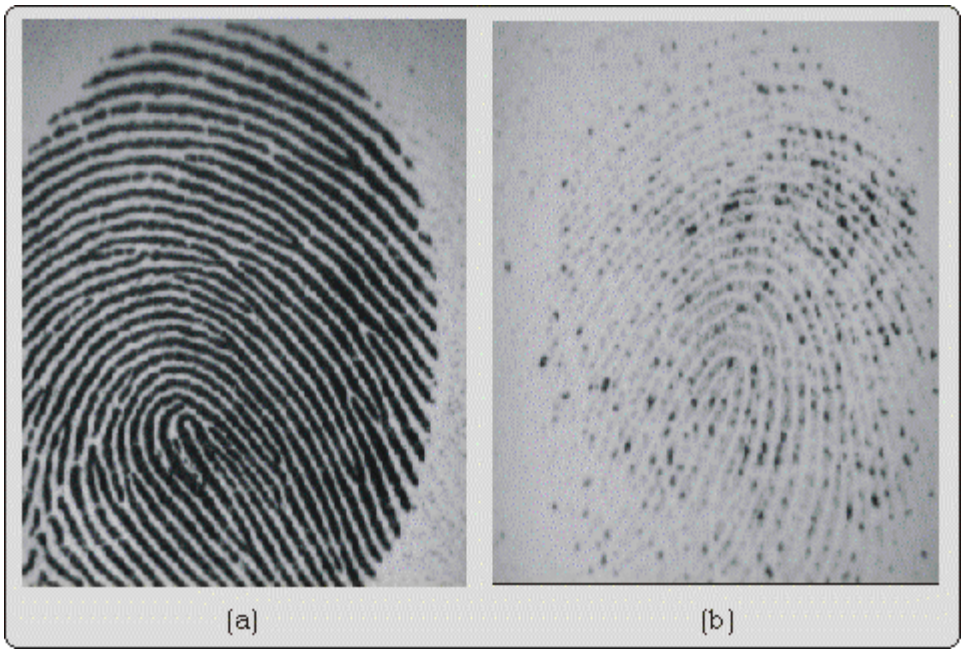


Figura A.5 Impresiones dactilares con diferente calidad intrínseca (a) buena calidad; (b) mala calidad.

A.3. Técnicas de adquisición

En el modo *off-line* se utilizan técnicas que involucran sustancias químicas, el caso típico es la impresión obtenida en papel de una huella dactilar entintada (*ink and paper*). En el modo *on-line* se utilizan técnicas que involucran sensores biométricos, comúnmente conocidas como tecnicas *live-scan*.

A.3.1. Basadas en sustancias químicas

Estas técnicas usualmente funcionan de la siguiente manera: una sustancia **A** es aplicada sobre el dedo, luego la yema es apoyada sobre una superficie **B** que reacciona con la sustancia **A**, logrando que aparezca en ésta la impresión dactilar. La técnica más conocida y utilizada en unidades forenses y policiales emplea la tinta como la sustancia aplicada y el papel como la superficie, con la peculiaridad que el dedo debe ser rolado (o rodado, ver figura A.6.a) de un lado a otro de tal manera que la impresión dactilar resultante no presente borrones o

manchas. En la práctica, esta técnica no es la más adecuada, puesto que un exceso de tinta puede manchar una parte o la impresión completa, mientras que una deficiencia de tinta producirá una impresión borrosa, afectando en gran medida su calidad y por consiguiente generando errores en el proceso de comparación [Coetzee y Botha, 1993].

Otras técnicas utilizan sustancias que no manchan la piel y originan una reacción sobre la superficie en la que el dedo es apoyado. Un caso especial utilizado en aplicaciones forenses es la obtención de impresiones dactilares latentes (ver figura A.6.b) sobre los objetos, como las que se dejan al sujetar una copa de cristal. La captura se realiza mediante polvos químicos que se adhieren a la impresión dactilar latente, siendo ésta recuperada mediante una película adhesiva o a través de una iluminación especial y fotografía.

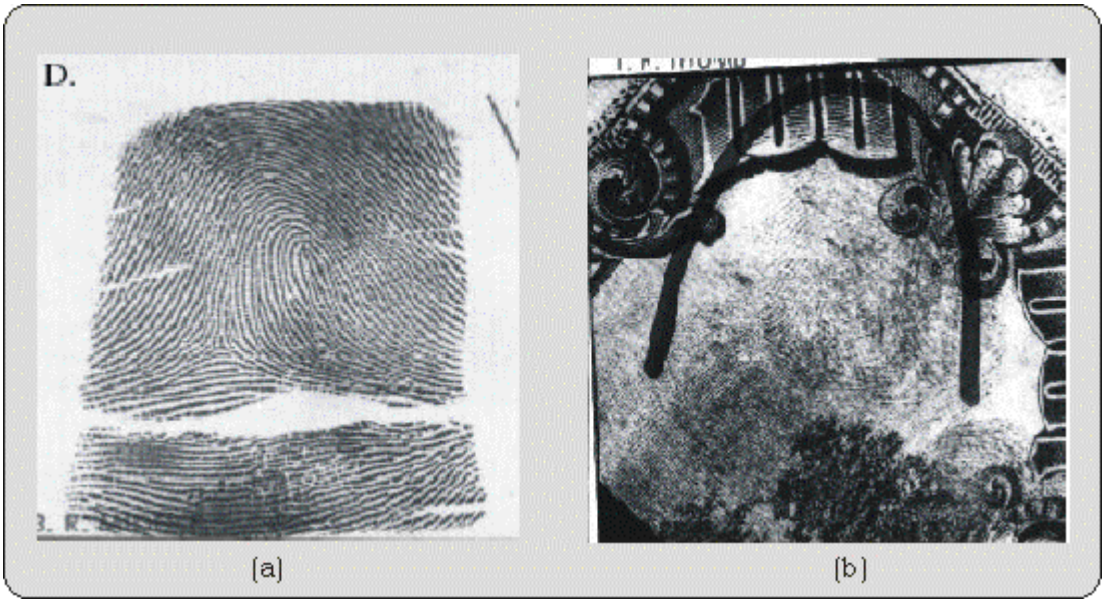


Figura A.6 (a) impresión dactilar rolada (b) impresión latente.

A.3.2. Basadas en sensores biométricos

Se utiliza un sistema electrónico de adquisición de datos denominado lector biométrico y por esta razón son mucho más eficientes que las basadas en sustancias químicas. El lector biométrico transforma los aspectos físicos extraídos

a una imagen digital (ver figura A.7). Casi todos los sensores existentes pertenecen a una de tres familias: los ópticos, de estado sólido y de ultrasonido.

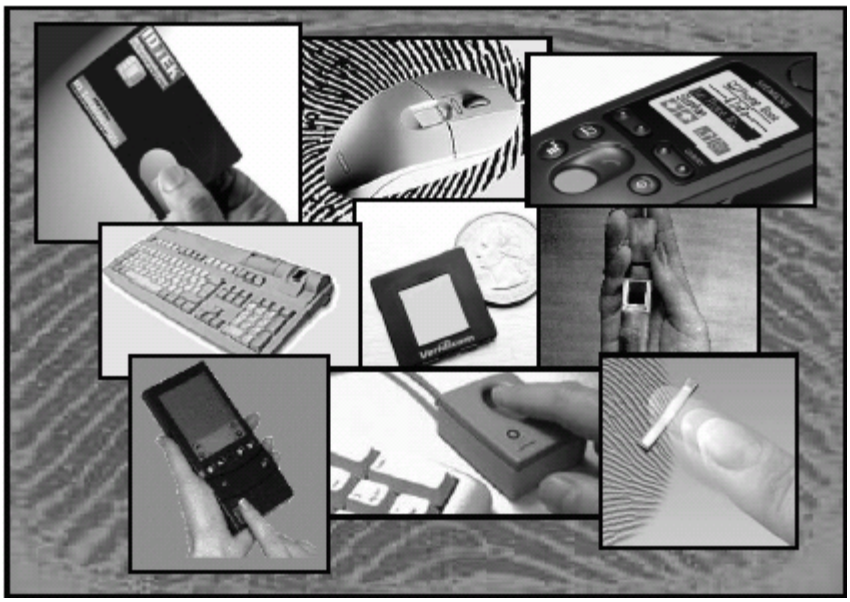


Figura A.7 Ejemplo de lectores biométricos electrónicos [Maltoni et al., 2003].

A.3.2.1. Sensores ópticos

La técnica de adquisición *live-scan* más antigua y utilizada es la basada en el principio de la Reflexión Interna Total Frustrada (*Frustated Total Internal Reflection* - FTIR), en donde el dedo es colocado sobre el lado de mayor área de un prisma de cristal, poniendo en contacto a las crestas con la superficie y dejando a los valles a una cierta distancia. La luz proveniente de una fuente difusa incide sobre uno de los menores lados del prisma ingresando en ángulo recto, reflejándose en las crestas y siendo absorbida por los valles, el patrón generado desde el otro lado menor incide en un lente y es capturado como una imagen analógica de video a través de un sensor que usa una tecnología de Dispositivos de Carga Acoplada (*Charge-Coupled Device* - CCD) o de Semiconductores de Óxido Metálico Complementario (*Complementary Metal-Oxide Semiconductor*, CMOS) siendo ésta digitalizada y enviada al computador a través de un grabador de cuadros (ver figura A.8).

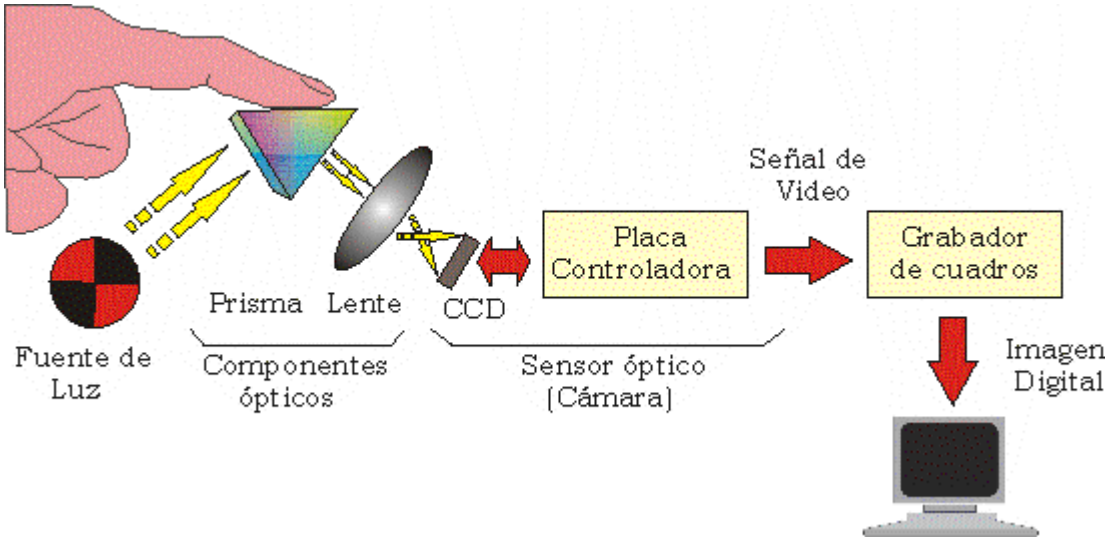


Figura A.8 Esquema de un lector óptico de huellas dactilares.

En la categoría de sensores ópticos también existen aquellos que utilizan una placa de pequeños prismas ubicados en forma adyacente que permiten disminuir un poco el tamaño del ensamble mecánico, sin embargo la calidad de las imágenes adquiridas generalmente es menor a los FTIR tradicionales.

Entre las principales ventajas de los sensores ópticos tenemos su resistencia a fluctuaciones de la temperatura, un costo relativamente bajo, la buena resolución, la buena calidad de la imagen y la posibilidad de mayores áreas de captura. Sin embargo, también poseen ciertas limitaciones, por ejemplo su tamaño no puede reducirse sustancialmente por las distancias focales requeridas y a menudo hay una gran distorsión en la imagen por la dificultad de enfocarla en un espacio tan pequeño. Además, se tiene el problema de la imagen latente, que aparece en el área de detección debido a la suciedad, la grasa y otras contaminaciones que presenta la huella; en un cierto período éstas degradan la capacidad de captura del dispositivo.

A.3.2.2. Sensores de estado sólido

Son conocidos también como sensores de silicio y se presentan como una alternativa para superar los inconvenientes en el tamaño y costo de los sensores ópticos, disminuyendo el área de captura. Sin embargo, un área grande de captura es parte de los requerimientos de un dispositivo de alta seguridad, esta necesidad incrementa los costos incluso para los de estado sólido.

Todos los sensores de silicio consisten de un arreglo de píxeles, donde cada píxel es en si un pequeño sensor. Cuatro diferentes métodos de captura de silicio han sido propuestos para convertir la información física en señales eléctricas: capacitivos, térmicos, de campo eléctrico y piezo-eléctricos [Maltoni et al., 2003].

A.3.2.3. Sensores de ultrasonido

En un sensor ultrasónico, un transmisor envía señales acústicas hacia la yema del dedo y un receptor detecta las señales del eco que rebotan de la superficie de la huella dactilar. La diferencia entre la impedancia acústica de la piel en las crestas y la impedancia del aire en los valles es usada para medir la distancia, adquiriendo de ese modo la impresión dactilar. El rango de frecuencias utilizadas por estos sensores varía desde 20 GHz a más. Son necesarias altas frecuencias para obtener la resolución requerida.

Se ha establecido que la calidad de la imagen mejorada de los sensores ultrasónicos produce tasas de precisión aproximadamente diez veces mejor que cualquier otra tecnología de captura en el mercado actual. La Tabla A.1 presenta las principales características de algunos sensores biométricos ofrecidos en el mercado.

	Tecnología	Compañía	Modelo	Dpi	Area (hxw)	Píxeles
Óptico	FTIR	Biometrika www.biometrika.it/eng/	FX2000	569	0.98"×0.52"	560×296 (165,760)
	FTIR	Digital Persona www.digitalpersona.com	UareU2000	440	0.67"×0.47"	316×228 (72,048)
	FTIR (sweep)	Kinetic Sciences www.kinetic.bc.ca	K-1000	up to 1000	0.002"×0.6"	2×900 (H×900)
	FTIR	Secugen www.secugen.com	Hamster	500	0.64"×0.54"	320×268 (85,760)
	Sheet prism	Identix www.identix.com	DFR 200	380	0.67"×0.67"	256×256 (65,535)
	Fiber optic	Delsy www.delsy.com	CMOS module	508	0.71"×0.47"	360×240 (86,400)
	Electro-optical	Ethentica www.ethentica.com	TactilSense T-FPM	403	0.76"×0.56"	306×226 (69,156)
Estado Sólido	Capacitive (sweep)	Fujitsu www.fme.fujitsu.com	MBF300	500	0.06"×0.51"	32×256 (H×256)
	Capacitive	Infineon www.infineon.com	FingerTip	513	0.56"×0.44"	288×224 (64,512)
	Capacitive	ST-Microelectronics us.st.com	TouchChip TCS1AD	508	0.71"×0.50"	360×256 (92,160)
	Capacitive	Veridicom www.veridicom.com	FPS110	500	0.60"×0.60"	300×300 (90,000)
	Thermal (sweep)	Atmel www.atmel.com	FingerChip AT77C101B	500	0.02"×0.55"	8×280 (H×280)
	Electric field	Authentec www.authentec.com	AES4000	250	0.38"×0.38"	96×96 (9,216)
	Piezoelectric	BMF www.bm-f.com	BLP-100	406	0.92"×"0.63	384×256 (98,304)

Tabla A.1 Principales características de algunos escáneres comerciales, agrupados por tecnología [Maltoni et al., 2003].

Anexo B
Filtros Bidimensionales

B.1. Introducción

En general un filtro bidimensional es una ley que permite la correspondencia de una imagen **I** con otra imagen **O**. Matemáticamente es representada por una transformación (u operador) que asigna a la entrada **I** una salida o respuesta **O** (ver figura B.1).

$$\begin{aligned} &\mathbf{I}(u,v) \xrightarrow{\Gamma} \mathbf{O}(u,v) \\ &\mathbf{O}(u,v) = \Gamma\{\mathbf{I}(u,v)\} \end{aligned} \qquad (B.1)$$

Entonces un filtro requiere una señal de entrada y produce una señal de salida, la cual está relacionada con la entrada a través de la transformación del sistema.

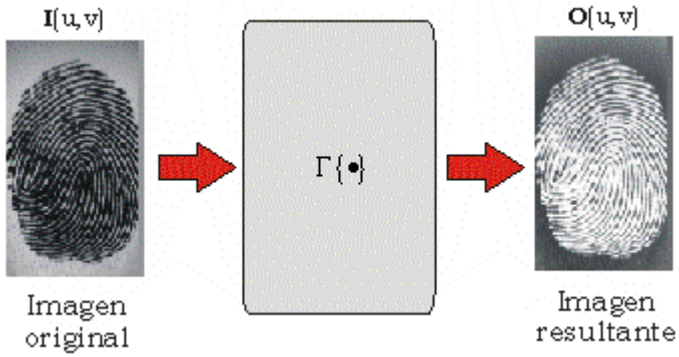


Figura B.1 Acción de un filtro sobre la imagen de entrada **I**.

El operador $\Gamma\{\bullet\}$ es la regla que define la acción del filtro sobre la imagen de entrada.

B.2. Propiedades de los filtros

1. **Linealidad:** Un operador es lineal si cumple con el "Principio de Superposición". Sean **F** y **Q** dos imágenes, a y b dos constantes arbitrarias, entonces el operador $\Gamma\{\bullet\}$ es lineal si y sólo si:

$$\Gamma\{a.\mathbf{F} + b.\mathbf{Q}\} = a.\Gamma\{\mathbf{F}\} + b.\Gamma\{\mathbf{Q}\} \tag{B.2}$$

2. **Invarianza ante desplazamientos:** Se dice que un operador es invariante ante desplazamientos, si un desplazamiento en una imagen de entrada resulta en un desplazamiento idéntico de la imagen de salida. Sean ${}^{nm}E\{\bullet\}$ un operador desplazamiento y **Q** una imagen, tal que:

$${}^{nm}E\{\mathbf{Q}(u,v)\} = \mathbf{Q}(u-n, v-m) \tag{B.3}$$

Entonces un operador es invariante ante desplazamientos si y sólo si, se cumple:

$$\Gamma\left\{{}^{nm}E\{\mathbf{Q}(u,v)\}\right\} = {}^{nm}E\left\{\Gamma\{\mathbf{Q}(u,v)\}\right\} \tag{B.4}$$

El operador de desplazamiento ${}^{nm}E\{\bullet\}$, produce una imagen desplazada n píxeles a la derecha y m píxeles hacia abajo respecto a su imagen de entrada.

3. **Respuesta impulsiva**: Todo filtro lineal invariante ante desplazamientos, puede ser caracterizado por su respuesta a una secuencia impulsiva **δ**, definida por:

$$\boldsymbol{\delta}(u,v) = \begin{cases} 1 & u = 0, v = 0 \\ 0 & \text{caso contrario} \end{cases} \tag{B.5}$$

La respuesta impulsional se definirá como:

$$\mathbf{W}(u,v) = \Gamma\{\boldsymbol{\delta}(u,v)\} \tag{B.6}$$

B.3. Convolución

La imagen **I** de tamaño $N_I x M_I$ se puede representar a través de una sumatoria de funciones impulsivas como se muestra a continuación:

$$\mathbf{I}(u,v) = \sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).\boldsymbol{\delta}(u-n, v-m) \tag{B.7}$$

De la ecuación (B.3) se deduce que: ${}^{nm}E\{\boldsymbol{\delta}(u,v)\} = \boldsymbol{\delta}(u-n, v-m)$, entonces la ecuación (B.7) podrá expresarse como:

$$\mathbf{I}(u,v) = \sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).{}^{nm}E\{\boldsymbol{\delta}(u,v)\} \tag{B.8}$$

Apliquemos a la imagen $\mathbf{I}(u,v)$ la acción de un filtro cuyo operador es $\Gamma\{\bullet\}$:

$$\begin{aligned}\mathbf{O}(u,v) &= \Gamma\{\mathbf{I}(u,v)\} \\ &= \Gamma\left\{\sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).{}^{nm}E\{\boldsymbol{\delta}(u,v)\}\right\} = \sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).{}^{nm}E\{\Gamma\{\boldsymbol{\delta}(u,v)\}\}\end{aligned} \tag{B.9}$$

Reemplazando (B.3) y (B.6) en (B.9) obtenemos:

$$\mathbf{O}(u,v) = \Gamma\{\mathbf{I}(u,v)\} = \sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).\mathbf{W}(u-n, v-m) \tag{B.10}$$

La operación expresada en la ecuación (B.10) es conocida como la operación de convolución y se denota en forma especial a través de:

$$\mathbf{I}(u,v) * \mathbf{W}(u,v) = \sum_{n=0}^{N_I-1} \sum_{m=0}^{M_I-1} \mathbf{I}(n,m).\mathbf{W}(u-n, v-m) \tag{B.11}$$

B.4. Máscaras de convolución espacial

Una máscara de convolución espacial es la representación discreta de la respuesta impulsional $\mathbf{W}(u,v)$, la cual es expresada como una matriz de coeficientes de tamaño NxM.

$$\mathbf{W} = \begin{bmatrix} \mathbf{W}(0,0) & \mathbf{W}(0,1) & \cdots & \mathbf{W}(0,M-1) \\ \mathbf{W}(1,0) & \mathbf{W}(1,1) & \cdots & \mathbf{W}(1,M-1) \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{W}(N-1,0) & \mathbf{W}(N-1,1) & \cdots & \mathbf{W}(N-1,M-1) \end{bmatrix} \qquad (B.12)$$

Esta representación de la máscara de convolución es conocida como filtro FIR bidimensional o simplemente filtro bidimensional.

Podremos reformular la ecuación (B.11) aprovechando que habrán coeficientes de la máscara $\mathbf{W}$ que son iguales a cero, veamos:

$$\begin{aligned} \mathbf{O}(u,v) = {} & \mathbf{I}(u,v).\mathbf{W}(0,0) + \cdots + \mathbf{I}(u,v+N-1).\mathbf{W}(0,N-1) + \\ & \mathbf{I}(u+1,v).\mathbf{W}(1,0) + \cdots + \mathbf{I}(u+1,v+M-1).\mathbf{W}(1,M-1) + \\ & \cdots\cdots\cdots\cdots \\ & \mathbf{I}(u+N-1,v).\mathbf{W}(N-1,0) + \cdots + \mathbf{I}(u+N-1,v+M-1).\mathbf{W}(N-1,M-1) \end{aligned}$$

$$\mathbf{O}(u,v) = \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \mathbf{I}(u+n,v+m).\mathbf{W}(n,m) \qquad (B.13)$$

Puede apreciarse que el número de operaciones a realizar será: N.M multiplicaciones y (N-1).(M-1) sumas, además la posición del resultado en la matriz de salida $\mathbf{O}(u,v)$ corresponderá con la ubicación de la esquina superior izquierda de la máscara, tal como muestra la figura B.2.

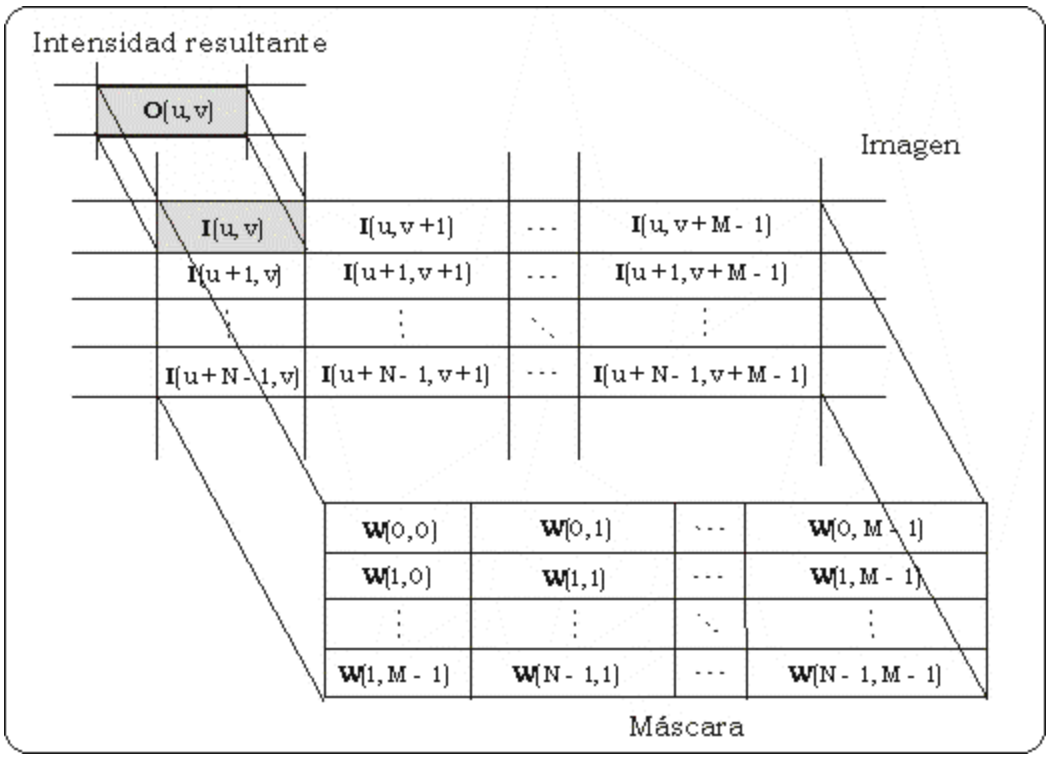


Figura B.2 Correspondencia entre los píxeles de **O** y **W** según la relación (B.13).

Si la posición del resultado en la matriz de salida $\mathbf{O}(u,v)$ corresponde con la posición (a,b) de **W** (ver figura B.3); la operación de convolución tomará la forma:

$$\mathbf{O}(u,v) = \sum_{u'=u-a}^{u+(N-1-a)} \sum_{v'=v-b}^{v+(M-1-b)} \mathbf{I}(u',v').\mathbf{W}(u'-u+a, v'-v+b) \qquad (B.14)$$

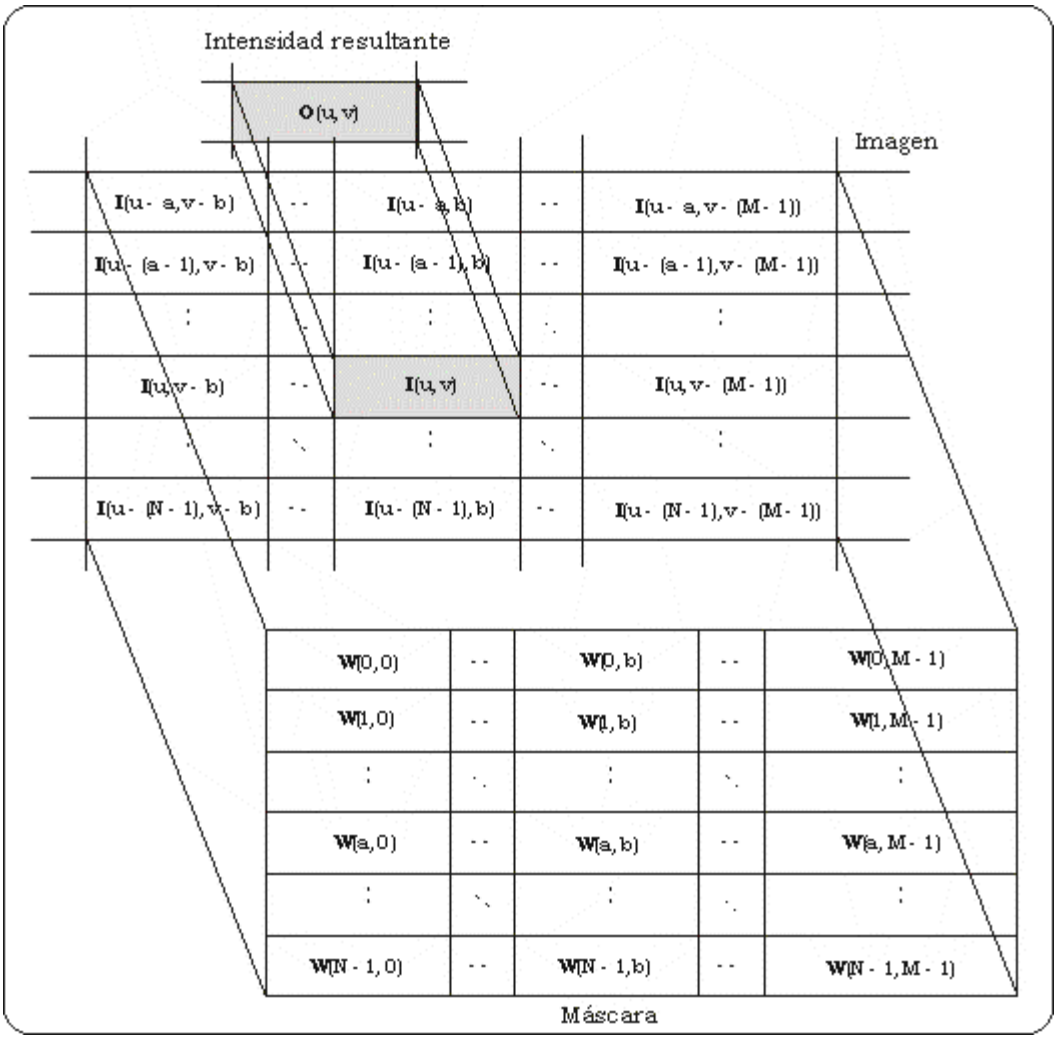


Figura B.3 Correspondencia entre los píxeles de **O** y **W** según la relación (B.14).

La posición usual para el par coordenado (a,b) es el centro de **W**, es decir: $a = \lfloor N/2 \rfloor = \overline{N}$ y $b = \lfloor M/2 \rfloor = \overline{M}$, bajo esta perspectiva es común suponer que el eje coordenado de los índices de la matriz **W** tienen como posición inicial el centro de la máscara por lo que los dominios para los índices son: $[-\overline{N}, +\overline{N}]x[-\overline{M}, +\overline{M}]$, pudiendo expresarse (B.14) como:

$$\mathbf{O}(u,v) = \mathbf{I}(u,v) * \mathbf{W}(u,v) = \sum_{u'=u-\overline{N}}^{u+\overline{N}} \sum_{v'=v-\overline{M}}^{v+\overline{M}} \mathbf{I}(u',v').\mathbf{W}(u'-u,v'-v) \qquad (B.15)$$

Cuando situamos a **W** en los bordes de **I** existen elementos de **W** que no corresponden con elementos de **I**. En este caso existen varias posibles soluciones:

1. Suponer que no son necesarios los resultados sobre los bordes y por lo tanto en la imagen **O** tomarán un valor fijo, por ejemplo cero. (ver figura B.4.a)
2. Suponer que en la imagen **I** se extienden los valores de los bordes, en este caso los puntos que están fuera de **I**, corresponden con los valores de cada borde. (ver figura B.4.b)
3. Suponer que la imagen **I** es periódica y por lo tanto se repite de manera infinita. En este caso los puntos que están fuera de **I** se pueden corresponder con valores en el otro extremo de ésta. (ver figura B.4.c)
4. Suponer que la imagen **I** está reflejada cuando salimos de los bordes. Esta quizá sea la mejor aproximación para convolucionar los píxeles de la periferia de una imagen aunque se sigue cometiendo un pequeño error de cálculo. (ver figura B.4.d)



Figura B.4 Posibles soluciones al efecto del filtrado de **I**: (a) se asume un valor constante en los bordes; (b) se reproduce el valor de los bordes; (c) se asume que **I** es periódica; (d) se asume que **I** está reflejada.

Desde un punto de vista teórico, para imágenes y filtros de gran tamaño resulta ideal realizar la convolución en el dominio de la frecuencia, desde un pun-

to de vista práctico puede ser mejor recurrir a la convolución espacial, pues es más rápida cuando tenemos máscaras de convolución pequeñas.

B.5. Máscaras de convolución discretas a partir de funciones continuas

Para poder filtrar una imagen de acuerdo a una función continua expresada en forma analítica se realiza un muestreo de dicha función. Este muestreo consiste en obtener el valor de la función en un conjunto discreto de puntos que representan la máscara de convolución **W**. Son dos las funciones analíticas de más interés estudiadas en este trabajo, las funciones de Gauss y Gabor bidimensionales por sus características de selectividad y por su simple sintonización. Veamos a continuación cada una de estas funciones y sus respectivas máscaras.

B.5.1. Máscara de Gauss bidimensional

Una función Gaussiana bidimensional (ver figura B.5) viene dada por la expresión:

$$W_{GS}(x, y; \sigma) = \frac{1}{2\pi\sigma^2} \exp\left\{\frac{1}{2\sigma^2}(x^2 + y^2)\right\} \qquad (B.16)$$

Donde:

(x, y): Son las coordenadas espaciales.

σ: Es la desviación estándar de la función Gaussiana. Un valor pequeño genera una función muy estrecha con un pico muy pronunciado mientras que un valor elevado produce una función con una pendiente muy suave.

El término $1/2\pi\sigma^2$ sirve para que el área total de la función sea siempre uno. Forzar a que esta área sea uno garantiza que tras el filtrado los valores obtenidos permanecen en el mismo rango que los valores de la imagen original.

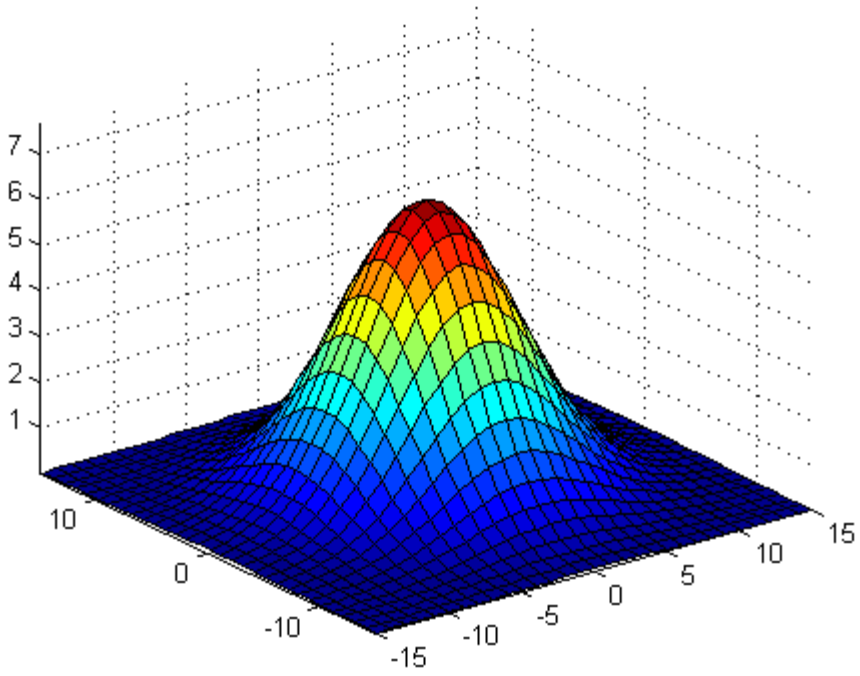


Figura B.5 Máscara de Gauss bidimensional considerando N = M = 31 y $\sigma = 6$.

El criterio habitual para determinar el tamaño de la máscara discreta es tomar un ancho de filtro tal que cubra la mayor parte central del área de la función. Por ejemplo, tomando $N = M = 5\sigma$ nos aseguramos de que muestreamos la función cubriendo un área del 98.76%. Otros autores consideran $N = M = 8\sigma$. Puesto que las máscaras de convolución suelen tener un número impar de elementos, tomaremos el ancho del mismo como $N = M = \lfloor 8\sigma + 1 \rfloor$. Tamaños superiores para las máscaras hacen que los cálculos sean muy costosos a cambio de no mejorar significativamente los resultados. Tamaños menores de la máscara aceleran los cálculos pero empeoran los resultados.

B.5.2. Máscara de Gabor bidimensional

Los filtros de Gabor simétricos tienen la siguiente forma general en el dominio espacial:

$$W_{GB}(x, y; \sigma_x, \sigma_y, \lambda, \theta) = \exp\left\{-\frac{1}{2}\left[\frac{x'^2}{\sigma_x^2} + \frac{y'^2}{\sigma_y^2}\right]\right\}\cos(\frac{2\pi}{\lambda}.x') \quad (B.17)$$

$$x' = x.sen(\theta) + y.cos(\theta) \quad (B.18)$$

$$y' = x.cos(\theta) - ysen(\theta) \quad (B.19)$$

Donde:

(x, y): Son las coordenadas espaciales.

(x', y'): Son las coordenadas espaciales rotadas.

σ_x y σ_y: Son las desviaciones estándares de la función Gaussiana.

λ: Es la longitud de onda. Especifica el período del componente cosenoidal de la función de Gabor. El rango de valores aceptables para λ está entre 2 y 256 pixeles.

θ: Especifica la orientación de la normal a las bandas paralelas de la función de Gabor. El rango de valores aceptables para θ está entre 0° y 180°.

Si realizamos el siguiente cambio de variables: $\sigma_x = \sigma$ y $\sigma_y = \frac{\sigma}{\gamma}$, obtendremos la definición alterna de la función de Gabor bidimensional:

$$W_{GB}(x, y; \sigma, \gamma, \lambda, \theta) = \exp\left\{-\frac{x'^2 + \gamma.y'^2}{2\sigma^2}\right\}\cos(\frac{2\pi}{\lambda}.x') \quad (B.20)$$

Donde:

γ: Es la razón de aspecto y determina la forma de la envolvente Gaussiana, (simetría circular, si $\gamma = 1$ y forma elíptica, en otro caso); los valores típicos se hallan entre 0.2 y 1.

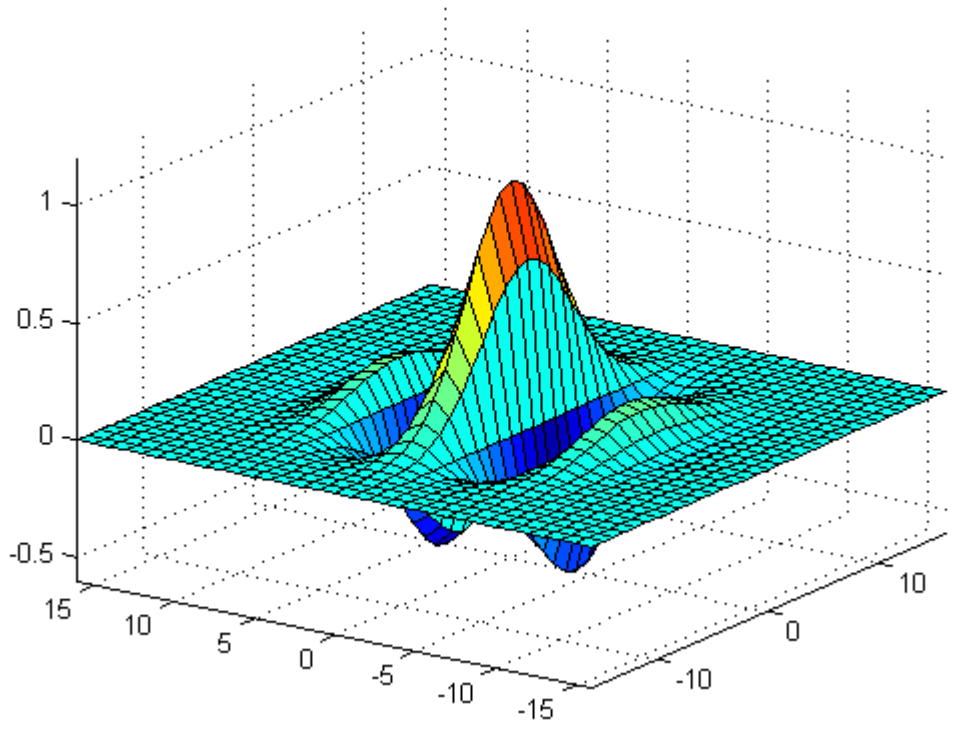


Figura B.6 Máscara de Gabor bidimensional considerando N = M = 33 y $\sigma_x = 4$, $\sigma_y = 4$, $\theta = 0^\circ$ y $\lambda = 8$.

La aplicación de los filtros de Gabor bidimensional sobre la imagen se realiza a través del uso de máscara de tamaño $N = M = \lfloor 8\sigma + 1 \rfloor$, donde se deberá especificar la orientación que tomará el filtro en cuestión, siendo definido por θ. Los parámetros del filtro son determinados por las relaciones obtenidas para σ_x, σ_y y λ. En figura B.6 mostramos una máscara de Gabor bidimensional, apreciando claramente las funciones exponenciales (para ambos ejes) moduladas respecto a la señal cosenoidal, pudiendo apreciarse las bandas de selección o detectores de líneas.

Anexo C
Vector Gradiente de una Imagen

C.1. Introducción

El cambio de iluminación sobre un píxel de la imagen respecto a sus vecinos está representado gráficamente por la pendiente que allí tiene la superficie definida por **I** y normalmente resulta diferente según sea la dirección considerada. La orientación del Vector Gradiente establece la dirección en la que el cambio es más intenso (apuntará hacia las zonas de mayor iluminación) y su magnitud es una medida de las pendientes (ver figura C.1). Por ejemplo, en un borde la magnitud del Vector Gradiente será alta y su dirección perpendicular al borde, debido a que allí la variación en la iluminación es drástica.

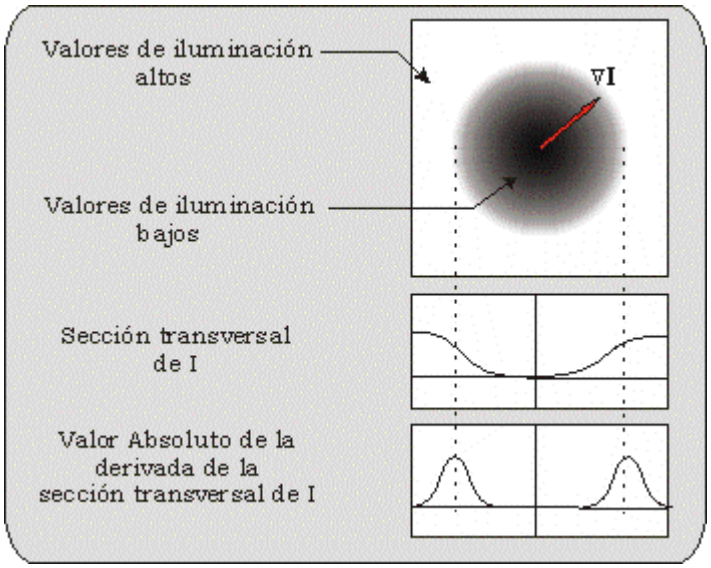


Figura C.1 Comportamiento del Vector Gradiente. Su dirección es un indicativo de la variación de la iluminación (desde las zonas de menor iluminación hacia las de mayor iluminación).

C.2. Representación discreta del Vector Gradiente

El Vector Gradiente sobre cada punto (u,v) de la imagen digital es obtenido a través del cálculo de las derivadas parciales respecto a los ejes x (columnas) e y (filas), tal como muestra la siguiente relación:

$$\nabla \mathbf{I}(u,v) = \begin{bmatrix} \dfrac{\partial \mathbf{I}(u,v)}{\partial x} \\ \dfrac{\partial \mathbf{I}(u,v)}{\partial y} \end{bmatrix} = \begin{bmatrix} \mathbf{G}_x(u,v) \\ \mathbf{G}_y(u,v) \end{bmatrix} \tag{C.1}$$

Donde, su magnitud y dirección vendrán dadas por las expresiones:

$$\left\| \nabla \mathbf{I}(u,v) \right\| = \sqrt{\mathbf{G}_x^2(u,v) + \mathbf{G}_y^2(u,v)} \tag{C.2}$$

$$\theta_V(u,v) = \tan^{-1}(\mathbf{G}_y(u,v) / \mathbf{G}_x(u,v)) \tag{C.3}$$

Es fundamental tomar en cuenta que al no ser una imagen digital una función continua requerirá utilizar representaciones discretas de las derivadas parciales que permitan el cálculo del Vector Gradiente; una de tales representaciones basada en diferencias finitas [De Oliveira, 2000, pp. 80] es:

$$\frac{\partial \mathbf{I}(u,v)}{\partial x} = \mathbf{I}(u, v+1) - \mathbf{I}(u,v) \tag{C.4}$$

$$\frac{\partial \mathbf{I}(u,v)}{\partial y} = \mathbf{I}(u+1, v) - \mathbf{I}(u,v) \tag{C.5}$$

Una variación útil de las ecuaciones definidas en (C.4) y (C.5), especializadas para el procesamiento de imágenes se da a través de las máscaras o filtros equivalentes a dichas ecuaciones mostradas en la figura C.2:

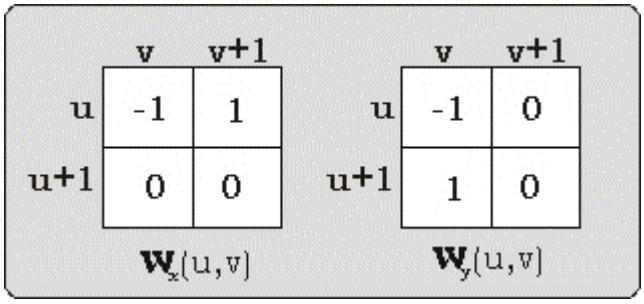


Figura C.2 Máscaras equivalentes a las derivadas parciales discretizadas de primer orden.

Las máscaras anteriormente citadas permiten determinar el Vector Gradiente de **I**, pero en la práctica debido al ruido presente en **I** y a la naturaleza de los filtros para imágenes, se han propuesto varias máscaras alternas, cada una con características específicas de rendimiento e inmunidad al ruido, entre las más destacadas tenemos:

C.2.1. Las Máscaras de Roberts

Son algunas de las más antiguas y de fácil cálculo. En vez de determinar el Vector Gradiente a través de la variación a lo largo de las direcciones x e y, éstas son detectadas en las direcciones 45° y 135° respecto al eje x. En este caso, las máscaras correspondientes son mostradas en la figura C.3.

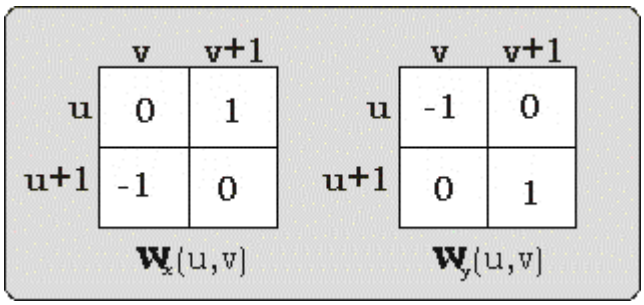


Figura C.3 Máscaras de Roberts.

Una gran desventaja de las máscaras de Roberts es su alta sensibilidad al ruido debido a los pocos píxeles empleados para determinar el Vector Gradiente.

C.2.2. Las máscaras de Prewitt y Sobel

Una forma de evitar que el Vector Gradiente sea calculado en un punto intermedio tal y como ocurre en las máscaras de Roberts, es usar entornos de tamaño 3x3 píxeles, como se muestra en la figura C.4.

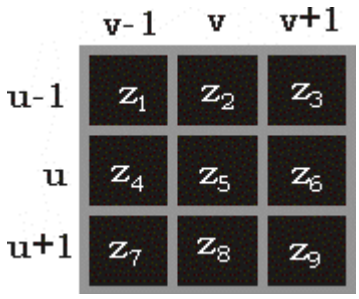


Figura C.4 Máscara 3x3.

Donde las variables $z_1, z_2, \ldots, z_9$ indican los valores de iluminación en una región de 3x3 píxeles de la imagen centrada en (u,v). La diferencia entre la tercera y la primera fila de la región de 3x3 píxeles provee una aproximación de la derivada en la dirección x, de igual forma la diferencia entre la tercera y la primera columna da una aproximación de la derivada en la dirección y. Las máscaras mostradas en la figura C.5 corresponden a las formuladas por Prewitt.

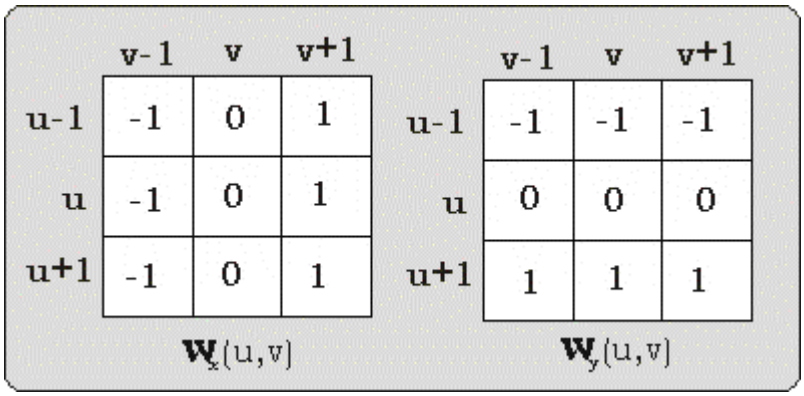


Figura C.5 Máscaras de Prewitt.

Las máscaras de Sobel (figura C.6) siguen la misma lógica que el caso anterior, pero dando mayor importancia a la variación de la intensidad que aparece en las líneas centrales (se duplican los coeficientes).

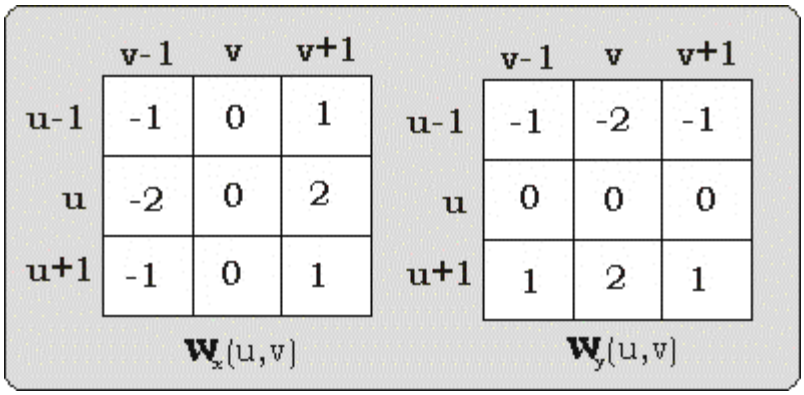


Figura C.6 Máscaras de Sobel.

Luego de establecer las máscaras $\mathbf{W}_x$ y $\mathbf{W}_y$ a emplear, procedemos a realizar la convolución, tal como se muestra en las siguientes relaciones:

$$\mathbf{G}_x(u,v) = \mathbf{I}(u,v) * \mathbf{W}_x(u,v) \qquad (C.6)$$

$$\mathbf{G}_y(u,v) = \mathbf{I}(u,v) * \mathbf{W}_y(u,v) \qquad (C.7)$$

Ambas máscaras deben ser de tamaño impar de forma tal que los operadores se encuentren centrados sobre los puntos en donde se calcula el Vector Gradiente. Para aquellos casos en donde las máscaras son de tamaño 2x2 píxeles (máscaras de Roberts), éstas deben ser expandidas a 3x3 píxeles a través del relleno de ceros.

En la figura C.7 se muestran los Vectores Gradiente calculados utilizando los tres tipos de máscaras descritas en la sección anterior.

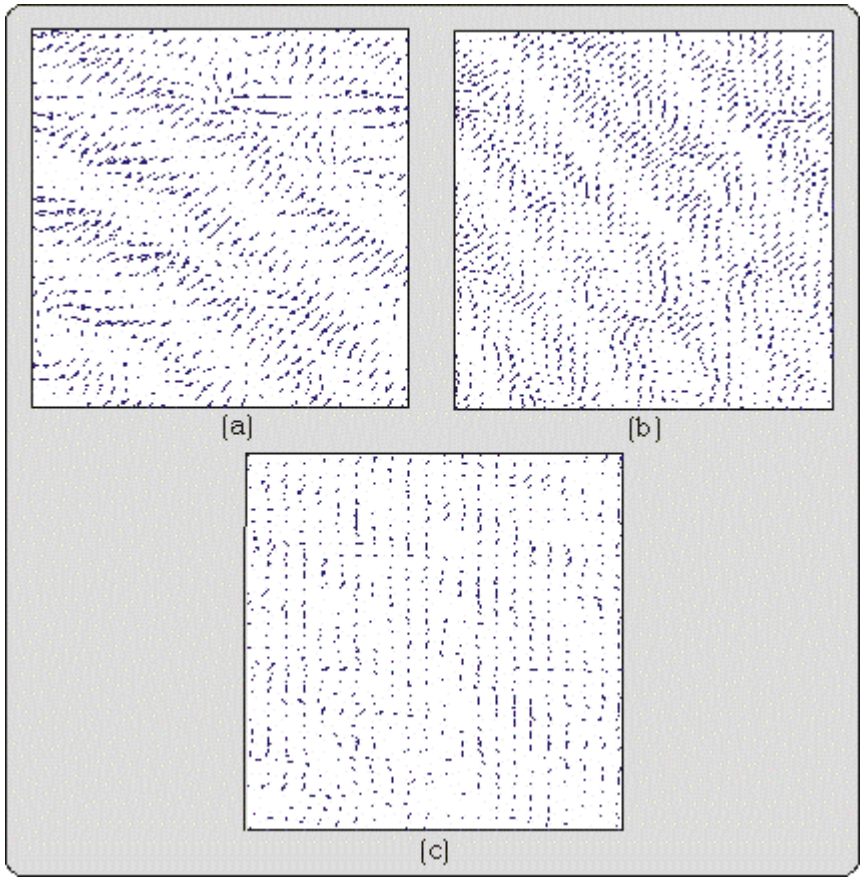


Figura C.7 Cálculo del Vector Gradiente para correspondientes porciones de cresta, a través de las máscaras de: (a) Sobel (b) Prewitt y (c) Roberts.

C.3 Información brindada por el Vector Gradiente de una imagen

El Vector Gradiente permite describir atributos importantes de una imagen como son: la homogeneidad y conformación estructural. Toda esta información viene resumida en la matriz de segundo momento y en la Matriz Hessiana, cada una de ellas particulariza cierto tipo de información brindada por el Vector Gradiente, como se esquematiza en la figura C.8.



Figura C.8 Información contenida en el Vector Gradiente y descriptores relacionados.

A continuación veamos cada uno de estos descriptores:

C.3.1. Matriz de Segundo Momento

La información acerca de la homogeneidad provista por el Vector Gradiente viene resumida en la matriz de segundo momento, también conocida como el tensor de estructura. Es un descriptor útil de la imagen para un gran número de problemas de bajo nivel en visión computacional, tal como la detección de características [Föstner y Gülch, 1987], estimación de formas [Kanatani, 1984], [Brown y Shvaytser, 1990], [Garding y Lindeberg, 1993], [Garding y Lindeberg, 1996] y análisis de movimiento [Bigün et al., 1991]. Este tipo de descriptor puede considerarse como la matriz de covarianza de una variable estocástica bidimensional, o como el momento de inercia de una distribución de masas en un plano. En nuestro caso determinaremos la matriz de segundo momento en el punto (u,v) de la imagen a través de la matriz de covarianza $\mathbf{C}_G$ de los pares del Vector Gradiente en dicho punto:

$$\mathbf{C}_G(u,v) = \begin{bmatrix} \mathbf{C}_{G_{xx}}(u,v) & \mathbf{C}_{G_{xy}}(u,v) \\ \mathbf{C}_{G_{xy}}(u,v) & \mathbf{C}_{G_{yy}}(u,v) \end{bmatrix} \tag{C.8}$$

Donde :

$$\mathbf{C}_{G_{xx}}(u,v) = \mathbf{G}_x^2(u,v) * \mathbf{W}_{GS}(u,v) \tag{C.9}$$

$$\mathbf{C}_{G_{yy}}(u,v) = \mathbf{G}_y^2(u,v) * \mathbf{W}_{GS}(u,v) \tag{C.10}$$

$$\mathbf{C}_{G_{xy}}(u,v) = \left\{\mathbf{G}_x(u,v).\mathbf{G}_y(u,v)\right\} * \mathbf{W}_{GS}(u,v) \qquad (C.11)$$

Son los estimados de la varianza y covarianza de $\mathbf{G}_x$ y $\mathbf{G}_y$, sobre la máscara Gaussina $\mathbf{W}_{GS}$ de tamaño $N_{W_{GS}} x N_{W_{GS}}$. Sabemos que $\mathbf{C}_G$ es simétrica definida positiva y por tanto existe una factorización:

$$\mathbf{C}_G = \mathbf{\Phi}_G.\mathbf{\Lambda}_G.\mathbf{\Phi}_G^T \qquad (C.12)$$

Donde $\mathbf{\Lambda}_G$ es la matriz correspondiente a los eingenvalores $\{\lambda_1, \lambda_2\}$ asumiendo que $\|\lambda_1\| \geq \|\lambda_2\|$ y $\mathbf{\Phi}_G$ es la matriz correspondiente a los eigenvectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_2$, tal que, para todo punto (u,v) se cumplirá que:

$$\lambda_{1,2} = \frac{1}{2}(\mathbf{C}_{G_{xx}}(u,v) + \mathbf{C}_{G_{yy}}(u,v)) \pm \frac{1}{2}\sqrt{(\mathbf{C}_{G_{xx}}(u,v) - \mathbf{C}_{G_{yy}}(u,v))^2 + 4\mathbf{C}_{G_{xy}}^2(u,v)} \qquad (C.13)$$

$$\boldsymbol{v}_{1,2} = \frac{1}{2}\begin{bmatrix} (\mathbf{C}_{G_{xx}}(u,v) - \mathbf{C}_{G_{yy}}(u,v)) \pm \sqrt{(\mathbf{C}_{G_{xx}}(u,v) - \mathbf{C}_{G_{yy}}(u,v))^2 + 4\mathbf{C}_{G_{xy}}^2(u,v)} \\ 2.\mathbf{C}_{G_{xy}}(u,v) \end{bmatrix} \qquad (C.14)$$

La matriz de segundo momento contiene información sobre la homogeneidad en una región (ver figura C.9) y puede ser clasificada como:

Isotrópica: Sus eigenvalores son prácticamente iguales y distintos de cero, lo que indica que la región no es constante ni existe alguna preferencia direccional del Vector Gradiente; esto generalmente corresponde a una región uniforme con presencia de ruido, donde la magnitud de los eigenvalores representa una medida de la variabilidad de la distribución del ruido.

Orientación dominante: El mayor de los eigenvalores es mucho más grande que los otros y su correspondiente eigenvector representa la dirección promedio del Vector Gradiente en esa región. Esta característica indica la existencia de un borde, el cual tiene orientación ortogonal al promedio del Vector Gradiente.

Constante: Ambos eigenvalores son cero, indicando que la región es constante.

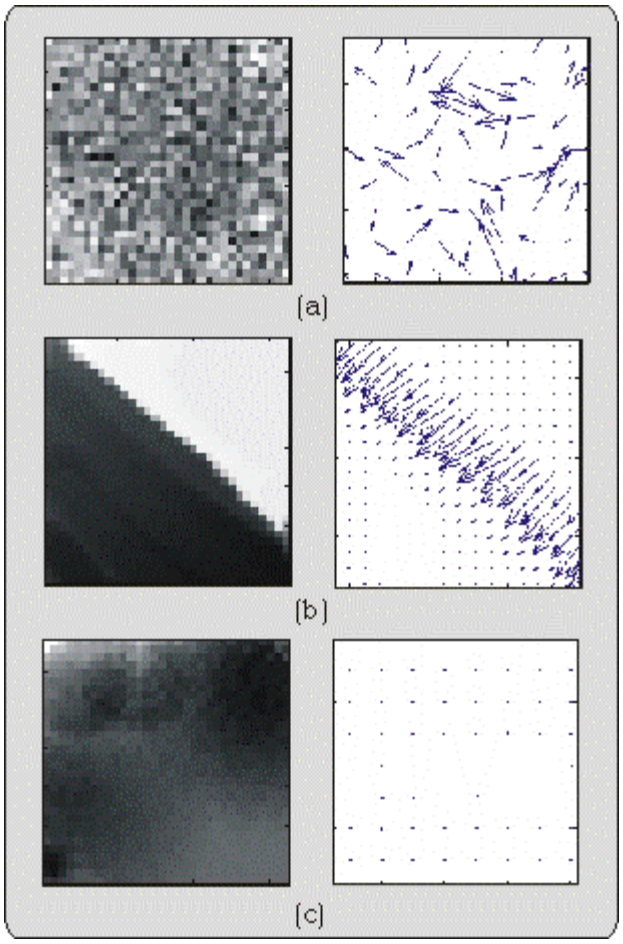


Figura C.9 Información de homogeneidad brindada por la matriz de segundo momento. (a) Región isotrópica $\lambda_1 \approx \lambda_2$;(b) Región con orientación dominante $\lambda_1 \gg \lambda_2$; (c) Región constante $\lambda_1 = \lambda_2 = 0$.

La información contenida en la matriz de segundo momento es resumida a través del indicador de anisotropía normalizada, también denominada coherencia, definida por la relación:

$$\mathrm{coh}[\mathbf{I}(u,v)] = \frac{\lambda_1 - \lambda_2}{\lambda_1 + \lambda_2} = \frac{\sqrt{(\mathbf{C}_{Gxx}(u,v) - \mathbf{C}_{Gyy}(u,v))^2 + 4\mathbf{C}_{Gxy}^2(u,v)}}{(\mathbf{C}_{Gxx}(u,v) + \mathbf{C}_{Gyy}(u,v))} \qquad (C.15)$$

La coherencia siempre estará en el rango de $\leq 0,1 \geq$. Si todos los Vectores Gradiente en la vecindad de tamaño $N_{W_G} x N_{W_G}$ apuntan exactamente en la misma dirección, la coherencia será igual a 1. Por otro lado, si los Vectores Gradiente en la vecindad de tamaño $N_{W_G} x N_{W_G}$ están igualmente distribuidos en todas las direcciones, la coherencia será igual a 0. Es decir, aquellas regiones con orientación dominante presentarán un valor de coherencia cercano a 1; mientras que las regiones constantes e isotrópicas tienden presentar una coherencia cercana a 0. Una manera de comprender intuitivamente el significado de la coherencia es a través de la representación gráfica de la matriz de segundo momento como una elipse:

$$\left\{ \boldsymbol{\xi} \in \square^2 : (\xi - (u,v))^T . \mathbf{C}_G(u,v) . (\boldsymbol{\xi} - (u,v)) = 1 \right\} \qquad (C.16)$$

Cuando la coherencia es baja, la elipse es aproximadamente circular y a medida que se incrementa se alargará; cuando la coherencia tiende a 1, la elipse degenera en una línea paralela a la dirección del Vector Gradiente. Una aplicación de interés para la coherencia es usarla como una medida que indique la presencia de minucias; puesto que, al ser las minucias discontinuidades en los patrones de crestas y valles de una impresión dactilar (una cresta/valle termina abruptamente) allí no presentará una orientación dominante, generando un nivel de coherencia bajo (ver figura 10). El inconveniente es que las áreas ruidosas también presentan niveles de coherencia bajos. Este problema puede ser resuelto en parte a través de la Imagen de Regiones (ver sección 3.3.4).

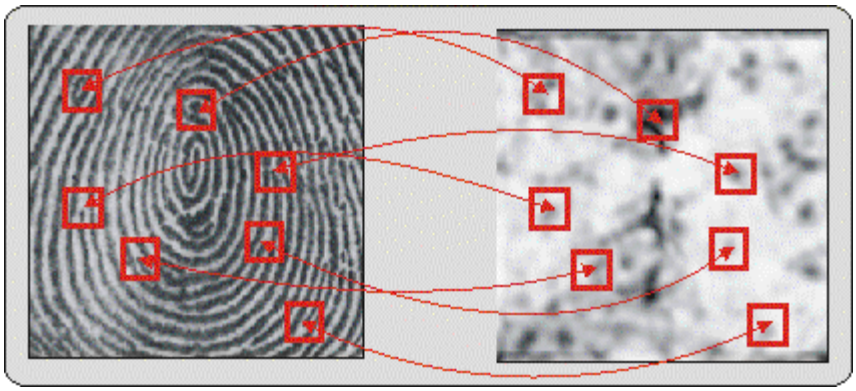


Figura C.10 Relación entre la coherencia y las posiciones de las minucias.

C.3.2. Matriz Hessiana

La Matriz Hessiana, resume las características estructurales que posee una imagen conformada por crestas y valles [Serrat et al., 2000], su uso para las impresiones dactilares es presentado en [Domeniconi, 1997], donde valiéndose de la Matriz Hessiana se obtiene la imagen adelgazada de la impresión dactilar.

Determinaremos la Matriz Hessiana en el punto (u,v) de la imagen a través de la aplicación del operador ∇ al Vector Gradiente, veamos:

$$\begin{aligned}\mathbf{M}_{\mathrm{H}}(u,v) = (\nabla^{\mathrm{T}}.\nabla)\mathbf{I}(u,v) &= \begin{bmatrix} \dfrac{\partial^2\mathbf{I}(u,v)}{\partial x^2} & \dfrac{\partial^2\mathbf{I}(u,v)}{\partial x\partial y} \\ \dfrac{\partial^2\mathbf{I}(u,v)}{\partial y\partial x} & \dfrac{\partial^2\mathbf{I}(u,v)}{\partial y^2} \end{bmatrix} \\ &= \begin{bmatrix} \mathbf{G}_{xx}(u,v) & \mathbf{G}_{xy}(u,v) \\ \mathbf{G}_{xy}(u,v) & \mathbf{G}_{yy}(u,v) \end{bmatrix}\end{aligned} \tag{C.17}$$

Las expresiones discretas de las derivadas parciales de orden superior a 1 se determinan por medio de manipulaciones adecuadas de la serie de Taylor para funciones de dos variables [De Oliveira, 2000, pp. 85-89], generándose las expresiones:

$$\frac{\partial^2\mathbf{I}(u,v)}{\partial x^2} = \mathbf{I}(u,v+1) - 2\mathbf{I}(u,v) + \mathbf{I}(u,v-1) \tag{C.18}$$

$$\frac{\partial^2\mathbf{I}(u,v)}{\partial y^2} = \mathbf{I}(u+1,v) - 2.\mathbf{I}(u,v) + \mathbf{I}(u-1,v) \tag{C.19}$$

$$\begin{aligned}\frac{\partial^2\mathbf{I}(u,v)}{\partial x\partial y} &= \frac{\partial^2\mathbf{I}(u,v)}{\partial y\partial x} \\ &= \frac{1}{4}(\mathbf{I}(u+1,v+1) - \mathbf{I}(u+1,v-1) + \mathbf{I}(u-1,v-1) - \mathbf{I}(u-1,v+1))\end{aligned} \tag{C.20}$$

Las máscaras o filtros equivalentes a las ecuaciones (C.18), (C.19) y (C.20), se muestran en la figura C.11.

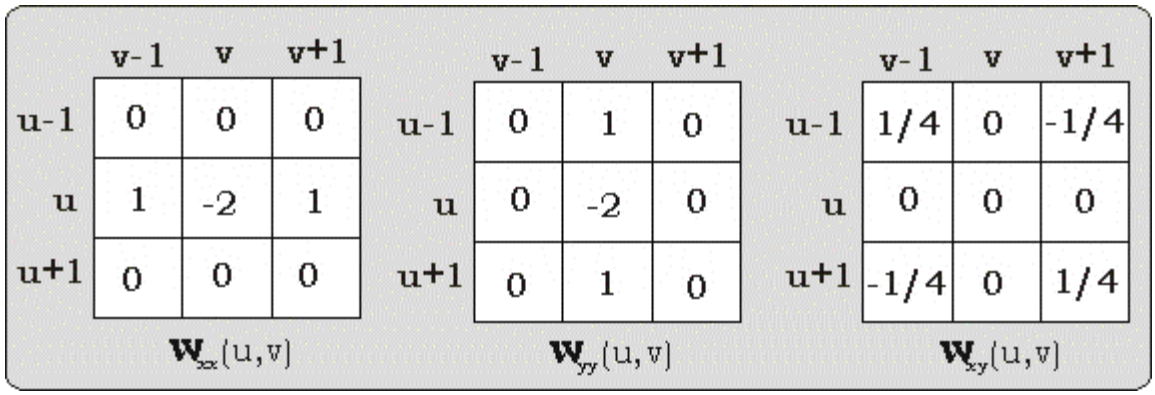


Figura C.11 Máscaras para determinar las derivadas parciales de segundo orden.

La forma habitual de obtener la Matriz Hessiana de una imagen $\mathbf{I}$ en un punto dado (u,v), es a través de la convolución de la imagen con las máscaras $\mathbf{W}_{xx}(u,v)$, $\mathbf{W}_{yy}(u,v)$ y $\mathbf{W}_{xy}(u,v)$, considerando que los bordes de la imagen se replican (ver sección B.4) :

$$\mathbf{G}_{xx}(u,v) = \mathbf{I}(u,v) * \mathbf{W}_{xx}(u,v) \tag{C.21}$$

$$\mathbf{G}_{yy}(u,v) = \mathbf{I}(u,v) * \mathbf{W}_{yy}(u,v) \tag{C.22}$$

$$\mathbf{G}_{xy}(u,v) = \mathbf{I}(u,v) * \mathbf{W}_{xy}(u,v) \tag{C.23}$$

Al ser también $\mathbf{M}_H$ simétrica definida positiva, presentará una factorización de la forma:

$$\mathbf{M}_H = \mathbf{\Phi}_H . \mathbf{\Lambda}_H . \mathbf{\Phi}_H^T \tag{C.24}$$

Donde $\mathbf{\Lambda}_H$ es la matriz correspondiente a los eigenvalores $\{\lambda_1, \lambda_2\}$ asumiéndose que $\|\lambda_1\| \geq \|\lambda_2\|$ y $\mathbf{\Phi}_H$ es la matriz correspondiente a los eigenvectores $\boldsymbol{v}_1$ y $\boldsymbol{v}_2$, tal que:

$$\lambda_{1,2} = \frac{1}{2}(\mathbf{G}_{xx}(u,v) + \mathbf{G}_{yy}(u,v)) \pm \frac{1}{2}\sqrt{(\mathbf{G}_{xx}(u,v) - \mathbf{G}_{yy}(u,v))^2 + 4\mathbf{G}_{xy}^2(u,v)} \tag{C.25}$$

$$\boldsymbol{v}_{1,2} = \frac{1}{2}\begin{bmatrix} (\mathbf{G}_{xx}(u,v) - \mathbf{G}_{yy}(u,v)) \pm \sqrt{(\mathbf{G}_{xx}(u,v) - \mathbf{G}_{yy}(u,v))^2 + 4\mathbf{G}_{xy}^2(u,v)} \\ 2.\mathbf{G}_{xy}(u,v) \end{bmatrix} \qquad (C.26)$$

Los eigenvectores son las direcciones en las cuales las segundas derivadas de **I** son maximizadas y los eigenvalores son los valores de cada máximo. Una de las aplicaciones de la Matriz Hessiana es determinar si un punto cualquiera de la imagen pertenece a una cresta o a un valle. Desde un punto de vista matemático una cresta es un punto para el cual los valores de intensidad asumen un mínimo local en la dirección de la curvatura principal, la condición de Saint-Venant (reformulada a través de los eigenvalores de la Matriz Hessiana) establece que si se cumplen las condiciones: $\|\lambda_1\| \geq \|\lambda_2\|$ y $\nabla \mathbf{I} // \boldsymbol{v}_2$ entonces:

si $\lambda_2 < 0$ entonces (u,v) pertenece a una cresta.

si $\lambda_2 > 0$ entonces (u,v) pertenece a un valle.

En la figura C.12 se muestra cómo a través del cumplimiento de la condición de Saint-Venant, se determinan aquellos puntos pertenecientes a una cresta, (puntos rojos sobre la impresión dactilar).

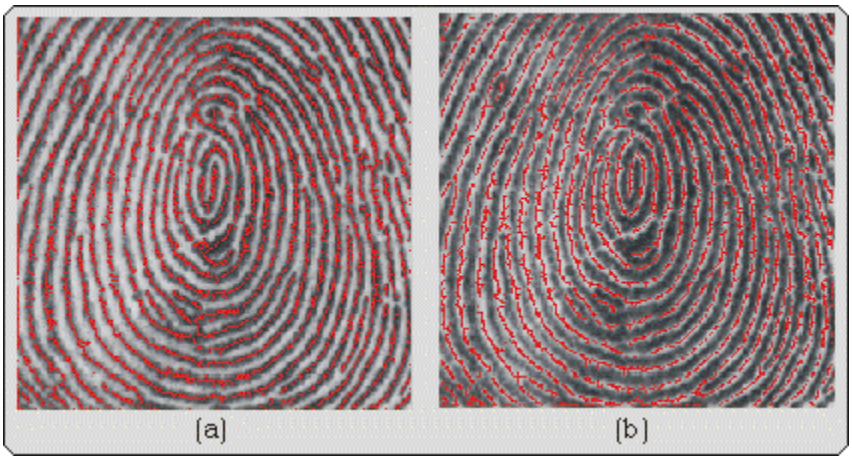


Figura C.12 Detección de crestas y valles (a) puntos vinculados a las crestas; (b) puntos vinculados a los valles.

Anexo D
Redes Neurales Artificiales

D.1. Introducción

Las Redes Neurales Artificiales (*Artificial Neural Networks* - ANN) son sistemas de procesamiento de información que poseen la capacidad de aprender, memorizar y generalizar a partir de un conjunto de datos y de su estructura. Se crearon como modelos muy abstractos y simples del sistema nervioso humano para comprender el proceso cognitivo, luego se analizaron como modelos matemáticos capaces de aprender de un conjunto de datos y con la llegada de los computadores se convirtieron en una poderosa herramienta computacional [Salas, 2002].

En los últimos años los modelos matemáticos, estadísticos y/o computacionales de las ANN, han sido aplicados como una herramienta alternativa para la resolución de problemas donde las técnicas clásicas han tenido dificultades. Algunas de las aplicaciones donde se han utilizado son en problemas de clasificación [Duda et al., 2001], aproximación de funciones [Del Brío y Sanz, 2002, pp. 64-66], pronóstico [Del Brío et al., 1995], reconocimiento de patrones [Pandya y Macy, 1996], control [Fukuda y Shibata, 1992], procesamiento de señales [Masters, 1994] y [Svarer, 1995] entre otras.

La clasificación de patrones determina a cual de las M clases pertenece un patrón de entrada con N elementos. Los clasificadores usualmente son entrenados por reglas de aprendizaje supervisado, en donde el número de clases es conocido y se asocian etiquetas de clase a los vectores de entrenamiento.

Los algoritmos de agrupamiento (*clustering*) auto-organizan y agrupan los datos de entrenamiento en M grupos (*clusters*). Después del entrenamiento, especifican a que grupo pertenecen los patrones de entrada. El agrupamiento o la cuantización de vectores se usan en aplicaciones de compresión y reconocimiento

de habla e imágenes. Los algoritmos de agrupamiento se entrenan por reglas de aprendizaje no supervisado usando datos de entrenamiento sin etiqueta.

Debido a la extensión de los temas relacionados con las ANN en este anexo se tratan sólo los modelos utilizados en el presente trabajo: el *Perceptron* como clasificador supervisado, además del *Self -Organizing Feature Map* (SOFM) como cuantificador óptimo de vectores. A continuación se presentan algunos conceptos importantes.

D.2. Paradigmas de aprendizaje

Se denomina paradigma de aprendizaje al modelo del entorno en el que la red neural trabaja e indica la forma en que el entorno influye en ese proceso de aprendizaje [Kung, 1993]. El paradigma de aprendizaje puede ser:

Supervisado. Se presentan los conocimientos en forma de pares de entrada y salida deseada. Si la entrada es distinta de la salida, se tendrá una red hetero-asociativa; si la entrada es igual que la salida, se tratará de una red auto-asociativa.

No supervisado. Durante este proceso de aprendizaje, a la red no se le presenta la salida deseada.

Por refuerzo. El instructor o maestro exterior sólo indica cuando la salida es correcta o no, pero no especifica en cuanto se diferencia de la salida deseada. Si se compara este paradigma con el supervisado, se observa que si bien el supervisado proporciona una información relativa a la dirección en la que se deben realizar los cambios en el sistema (ajuste de los pesos), en el caso de un aprendizaje por refuerzo no se tiene información acerca de la "dirección" del cambio, lo cual hace que su ámbito de aplicación sea mucho más reducido comparado con el modo supervisado.

Aprendizaje híbrido. Se trata de una combinación del aprendizaje supervisado y del no supervisado. Parte de los pesos se ajustan por medio de un esquema de aprendizaje supervisado y el resto se obtienen por medio de un aprendizaje no supervisado.

D.3. Protocolos de entrenamiento

Desde un punto de vista global, el entrenamiento supervisado consiste en presentar a una red aquellos patrones cuyas etiquetas se conocen, encontrar la salida de la red y ajustar sus parámetros internos para que la salida sea semejante a la deseada. Los cuatro protocolos útiles de entrenamiento supervisado [Duda et al., 2001, pp. 293-295] son: estocástico, por lotes (*batch*), en línea (*on-line*) y por bloques.

Estocástico: En donde los patrones (datos) son escogidos aleatoriamente del conjunto de entrenamiento y los parámetros de la red son actualizados por cada presentación del patrón. Este protocolo es llamado estocástico debido a que los datos de entrenamiento pueden ser considerados como variables aleatorias.

Por lotes: En donde todos los patrones son presentados a la red antes de que el proceso de aprendizaje se dé lugar, es decir que haya algún cambio en sus parámetros.

En línea: En donde cada patrón es presentado solamente una vez, aquí no se requiere de una memoria para almacenar los patrones.

Por bloques: En donde un subconjunto de patrones se presentan a la red y sus parámetros son actualizados para cada subconjunto hasta que se presente un bajo nivel de variación en los parámetros internos de la red.

La cantidad de presentaciones de los patrones se describe por el número de épocas, donde una época corresponde a una sola presentación de todos los patrones del conjunto de entrenamiento. El número de épocas es un indicador de la cantidad relativa de aprendizaje.

D.4. Modelo Biológico de una Neurona Real

Una de las características básicas de los modelos de ANN es la representación de una unidad de procesamiento elemental. La neurona se compone principalmente del cuerpo de la célula o soma de donde se desprende una serie de ramificaciones llamadas fibras nerviosas o dendritas y además contiene un núcleo

en su interior. Las dendritas constituyen la superficie de entrada de la neurona. Desde el cuerpo de la neurona surge una larga fibra llamada axón cuyas ramas forman la arborización axonal. Las terminaciones de las ramas del axón conocidas como nervios terminales o botones actúan sobre otras neuronas mediante un proceso conocido como sinapsis.

D.5. Modelo Matemático de una Neurona Real: El Perceptron

D.5.1 Perceptron para dos clases separables

El *perceptron* también conocido como nodo o neurona, es la unidad de procesamiento de información más simple. En esencia un *perceptron* dividirá el espacio de salida en dos regiones a través de un hiperplano de separación lineal, cuyo límite está basado en el umbral obtenido de la combinación lineal del vector de entrada con un vector de coeficientes. Para toda j-ésima observación $\mathbf{y}_j$ se definirá el vector aumentado de observaciones $\overline{\mathbf{y}}_j$ como: $\overline{\mathbf{y}}_j = [\mathbf{y}_j \; 1]^T$, este vector será multiplicado por otro vector conocido como vector de pesos $\mathbf{w}_p$ que determina la ecuación del hiperplano de separación lineal. El ultimo elemento de $\overline{\mathbf{y}}_j$ usualmente denominado *bias,* es utilizado para aumentar un grado de libertad al aprendizaje del clasificador, la clase asignada al vector aumentado de observaciones $\overline{\mathbf{y}}_j$ estará dada por la siguiente función de decisión:

$$\tilde{c}_j = \begin{cases} w_1 & \text{signo}(\overline{\mathbf{y}}_j * \mathbf{w}_p^T) > 0 \\ w_{-1} & \text{caso contrario} \end{cases} \qquad (D.1)$$

Entonces, esta función discrimina correctamente si $c_j = \tilde{c}_j$. En general, el clasificador es entrenado variando el vector de pesos de acuerdo con el error entre la clase calculada $\tilde{c}_j$ y la clase verdadera c_j; definida como:

$$e_j = c_j - \tilde{c}_j \qquad (D.2)$$

Encontrar un hiperplano óptimo que minimice el error no es un problema sencillo [Freeman y Skapura, 1993], puesto que durante el entrenamiento el clasificador estará influenciado por las observaciones de clases traslapadas y no lineales.

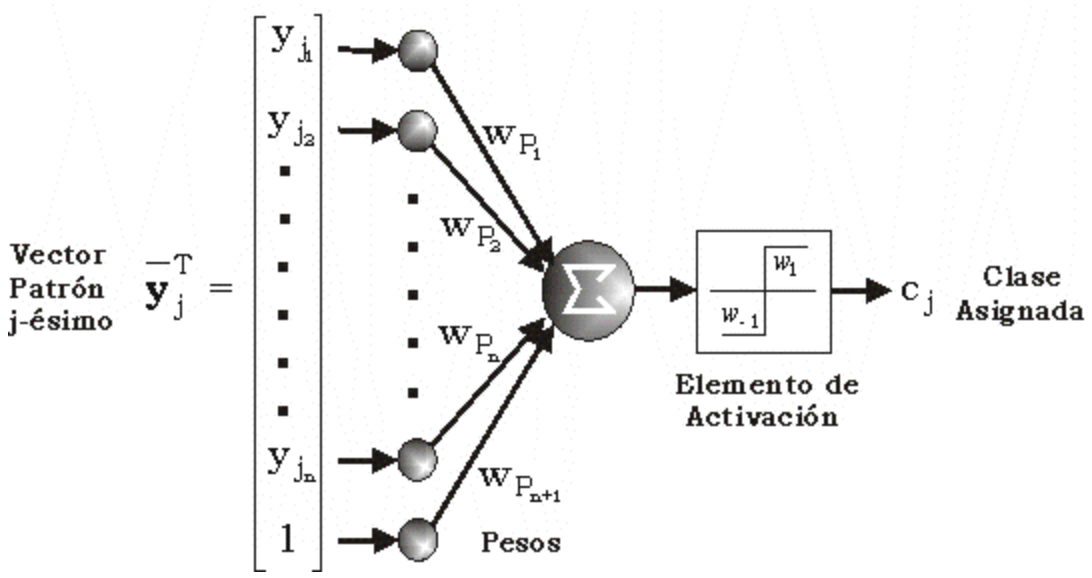


Figura D.1 Representación del *perceptron* para dos clases.

D.5.2. Regla de Aprendizaje Perceptron

El algoritmo original de aprendizaje por corrección de errores fue desarrollado por Rosenblatt. Este requiere que sus resultados sean evaluados y se realicen las oportunas modificaciones en sus pesos y umbrales (parámetros libres) si fuera necesario, en este caso el error es únicamente diferente de cero para vectores de entrada que son incorrectamente clasificados. Los pasos del algoritmo son.

1. Inicialización del vector de pesos $\mathbf{w}_p$.
2. Presentación de un nuevo par $(\mathbf{y}_j, c_j)$ del conjunto de entrenamiento.
3. Cálculo de la clase asignada $\tilde{c}_j$.
4. Si el error es diferente de cero, se actualiza el vector de pesos, de acuerdo con:

$$\mathbf{w}_p(t+1) = \mathbf{w}_p(t) + \alpha * e_j(t) * \overline{\mathbf{y}}_j^T \qquad (D.3)$$

Donde t es el paso de la iteración, α es un factor de ganancia en el rango de 0 a 1.0, que define la velocidad de aprendizaje.

5. Si el error no ha alcanzado un valor umbral predefinido, volver al paso 2.

El *perceptron* puede ser utilizado como máquina universal de aprendizaje [Hilera y Martínez, 1995, pp. 101-103]. Sin embargo provee convergencia solamente para dos clases linealmente separables: es decir que para problemas que no son linealmente separables este algoritmo no terminará en un número finito de iteraciones. Una forma común para superar el problema de la terminación del algoritmo es forzar a que se detenga estableciendo un número máximo de iteraciones permitidas (paso 5).

D.5.3. Regla de Aprendizaje Fuzzy Perceptron

El algoritmo de aprendizaje *Fuzzy Perceptron* propuesto por [Keller y Hunt, 1985] es una herramienta de gran utilidad, pues provee una forma elegante de tratar con el problema de la terminación del algoritmo y al mismo tiempo disminuye la influencia negativa producida por el traslape entre clases para espacios no linealmente separables, debido a que utiliza la teoría de conjuntos *Fuzzy* para obtener un indicador cuantitativo de incertidumbre sobre el grado de pertenencia a la clase verdadera, de tal forma que los vectores que presentan un alto grado de incertidumbre tengan menos influencia sobre los resultados del entrenamiento. Keller y Hunt establecieron la siguiente forma de asignar los valores de membresía *Fuzzy* para una bipartición.

$$u_1(\mathbf{y}_j) = \begin{cases} 0.5 + \dfrac{\exp(f(d_{-1}(\mathbf{y}_j) - d_1(\mathbf{y}_j))/d) - \exp(-f)}{2(\exp(f) - \exp(-f))} & \mathbf{y}_j \in \mathbf{Y}_1 \\ 1 - u_{-1}(\mathbf{y}_j) & \mathbf{y}_j \in \mathbf{Y}_{-1} \end{cases} \tag{D.4}$$

$$u_{-1}(\mathbf{y}_j) = \begin{cases} 0.5 + \dfrac{\exp(f(d_1(\mathbf{y}_j) - d_{-1}(\mathbf{y}_j))/d) - \exp(-f)}{2(\exp(f) - \exp(-f))} & \mathbf{y}_j \in \mathbf{Y}_{-1} \\ 1 - u_1(\mathbf{y}_j) & \mathbf{y}_j \in \mathbf{Y}_1 \end{cases} \tag{D.5}$$

Puesto que en este caso el problema de clasificación es de dos clases, se asume que: $u_1(\mathbf{y}_j) + u_{-1}(\mathbf{y}_j) = 1$.

Donde:

$u_k(\mathbf{y}_j) \in \leq 0,1 \geq$: Es la función de membresía que describe el grado en que la

j-ésima observación del conjunto de entrenamiento $S_{\mathbf{Y}}$ pertenece a la k-ésima clase.

$d_k(\mathbf{y}) = \left\| \mathbf{y} - \boldsymbol{\mu}_k \right\|$: Es la distancia entre el vector $\mathbf{y}$ y el vector promedio $\boldsymbol{\mu}_k$ de la k-ésima clase, siendo:

$$\boldsymbol{\mu}_k = \frac{1}{n_1} \sum_{j=1}^{n_c} \mathbf{Y}_{j:k} = [\mu_k^{Mean} \quad \mu_k^{Var} \quad \mu_k^{Coh}]^T \tag{D.6}$$

$d = \left\| \boldsymbol{\mu}_1 - \boldsymbol{\mu}_{-1} \right\|$: Es la distancia entre los dos vectores promedio de cada clase.

f: Es una constante que controla el índice en que los grados de membresía decrecen hacia 0.5.

El algoritmo de aprendizaje *Fuzzy Perceptron* consta de los siguientes pasos:

1. Inicialización del vector de pesos $\mathbf{w}_p$.
2. Presentación de un nuevo par $(\mathbf{y}_j, c_j)$ del conjunto de entrenamiento.
3. Cálculo de la clase asignada $\tilde{c}_j$.
4. Si el error es diferente de cero, se actualiza el vector de pesos, de acuerdo con:

$$\mathbf{w}_p(t+1) = \mathbf{w}_p(t) + \alpha * \left| u_1(\mathbf{y}_j) - u_{-1}(\mathbf{y}_j) \right|^m * e_k(t) * \overline{\mathbf{y}}_j^T \tag{D.7}$$

5. Si no se ha alcanzado el número máximo de iteraciones, volver al paso 2.

El parámetro α es la tasa de aprendizaje y m es una constante positiva que controla el grado de "incertidumbre", entonces la observación $\mathbf{y}_j$ más "incierta" tendrá menos influencia en la adaptación de los pesos. Esta técnica de entrenamiento puede interpretarse como una generalización del algoritmo de aprendizaje del *Perceptron*, puesto que si suponemos que no hay incertidumbre de las muestras de entrenamiento, entonces la expresión: $\left| u_1(\mathbf{y}_j) - u_{-1}(\mathbf{y}_j) \right|^m$ se reduce a 1, obteniéndose la regla de aprendizaje del *perceptron* estándar. Los pasos del entrenamiento del *Fuzzy Perceptron* son similares a los pasos 1 al 3 del *Perceptron* estándar, sin embargo el cuarto paso incluye el cálculo de los valores de membresía de las observaciones. Además el quinto paso, relacionado con la terminación del algoritmo incluye que el valor de membresía esté dentro del siguiente rango: $[0.5 - \xi, 0.5 + \xi]$ donde ξ es una constante determinada experimentalmente. Por lo

tanto, la condición adicional de terminación es que si los errores de entrenamiento son producidos por las muestras cuyos valores de membresía están dentro de este rango, entonces el algoritmo debería terminar. El parámetro ξ debe ser adecuadamente seleccionado, de tal forma que las muestras de entrenamiento cuyos valores de membresía que estén fuera del rango sean linealmente separables.

D.6. Consideraciones del aprendizaje

Control de Convergencia: Es conveniente avanzar por la superficie de error con incrementos pequeños de los pesos, debido a que sólo se tiene una información local de la superficie y no se sabe lo lejos o lo cerca que se está del punto mínimo. Por otro lado, con incrementos grandes, se corre el riesgo saltar el valor mínimo del error y estar oscilando alrededor de él, pero sin poder alcanzarlo. El parámetro que ajusta el valor del incremento es la tasa de aprendizaje.

Inicialización y cambio de pesos: Es necesario inicializar los valores de los pesos con valores aleatorios pequeños, usualmente normalizados dentro de un rango, el cual se elige de tal modo que el resultado de la suma de las entradas ponderadas por los pesos no sea muy grande y no sature la función de activación. Por esto usualmente se elige un rango entre [-0.5,0.5], o [0,1]. La modificación de los pesos puede realizarse cada vez que un patrón ha sido presentado o después de haber acumulado los cambios de los pesos en un número de iteraciones, el momento adecuado para el cambio de los pesos depende de la aplicación.

D.7. Mapas Auto-organizados

El mapa auto-organizado de características (*Self-Organizing Feature Map* - SOFM) permite el agrupamiento de espacios vectoriales y tiene la particularidad de imponer sobre los agrupamientos convencionales ciertas relaciones de vecindad entre las muestras o representantes de los grupos conseguidos, además es también utilizado para cuantificación óptima de vectores.

La arquitectura general de un SOFM (ver figura D.2) puede describirse de la siguiente manera: Trátese de una red de dos capas con N_{int} nodos en la capa de

entrada y N_{out} nodos en la capa de salida dispuestas en una malla usualmente conocida como *capa de competición*. La forma de la malla puede ser definida rectangular, hexagonal o incluso irregular. El i-ésimo nodo de la capa de competición esta conectado a cada nodo de la capa de entrada a través de N_{int} conexiones hacia adelante, implicando la asociación de un vector de pesos denotado por $\mathbf{U}_i = \begin{bmatrix} \mu_{i1} & \mu_{i2} & \cdots & \mu_{ij} & \cdots & \mu_{iN_{int}} \end{bmatrix}^T$, además se define alrededor de este nodo un conjunto de vecindad $\mathbf{N}_i$ que puede adoptar diversas formas, cuyo efecto es equivalente a conexiones laterales de inhibición implícitas (peso negativo), pues aunque no estén conectados, los nodos tendrán cierta influencia sobre sus vecinos. El valor que se asigne a los pesos $\mathbf{U}_i$ durante el proceso de aprendizaje de la red dependerá precisamente de esta interacción lateral. La entrada de información a la red es un vector $\mathbf{x}(t) = \begin{bmatrix} x_1 & x_2 & \cdots & x_{N_{int}} \end{bmatrix}^T$ que se conecta para el caso mas sencillo, en paralelo a la capa de entrada (puede haber otras configuraciones), note que el peso μ_{ij} es debido a la conexión entre el nodo j-ésimo de la capa de entrada y el nodo i-ésimo de la capa de competición. La red funciona de manera que dado un vector de entrada $\mathbf{x}$, se activa una única neurona en la capa de competición (representa el punto del espacio más próximo al vector de entrada). En consecuencia, cada nodo de la capa de competición tiene asociado un punto en el espacio de representación cuyas coordenadas vienen dadas por su vector de pesos. A estos puntos los denominaremos "prototipos" o también vectores de referencia.

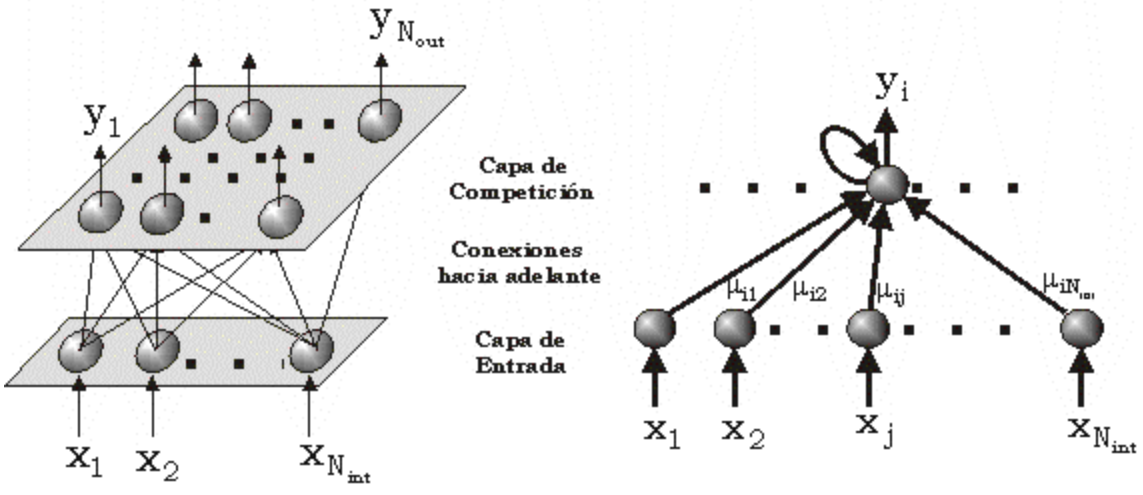


Figura D.2 Arquitectura del SOFM.

El algoritmo que actualiza los pesos $\mathbf{U}_i$ está compuesto por dos fases fundamentales: la presentación de las muestras de entrenamiento y la corrección de los pesos asociados a la vecindad del nodo activo, tal que se consiga una malla "ordenada" topológicamente, en la que cada nodo esté especialmente sensibilizado en un dominio de la señal de entrada. Los pasos del algoritmo son:

1. Inicialización de los pesos: En un principio $t = t_o$, la red se encuentra en un estado inicial desordenado, tal que los pesos $\mathbf{U}_i(t)$ toman valores aleatorios.

2. Presentación de la entrada: El vector de información $\mathbf{x}(t) \in \Re^{N_{int}}$ es pasado a la capa de entrada.

3. Búsqueda del nodo ganador: El vector de información $\mathbf{x}(t)$ es comparado con todos los vectores de pesos $\mathbf{U}_i(t)$, hasta encontrar el nodo c cuyo vector de pesos $\mathbf{U}_c(t)$ esté lo más cercano a $\mathbf{x}(t)$. Frecuentemente se utiliza la métrica Euclidiana, pero puede emplearse otros tipos de métrica.

$$\|\mathbf{x}(t) - \mathbf{U}_c(t)\| = \min_i \{\|\mathbf{x}(t) - \mathbf{U}_i(t)\|\} \qquad (D.15)$$

4. Definición de la vecindad: Definimos una vecindad alrededor del nodo ganador c, denotada por $N_c(t)$, ésta puede tener diferentes configuraciones (ver figura D.3) y cambia con el tiempo, tal que inicialmente incluye a todos los nodos de la capa de competición y a medida que el algoritmo transcurre decrece en forma lineal. En la fase de convergencia la vecindad contendrá al nodo ganador y los vecinos más próximos.

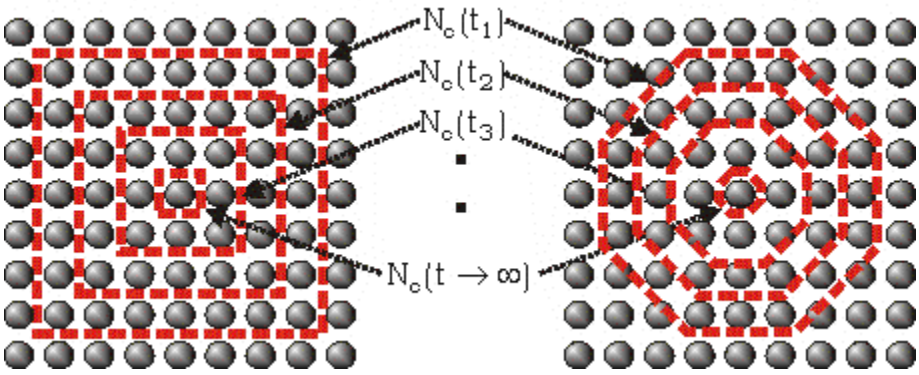


Figura D.3 Conjuntos de vecindad.

5. Modificación de los pesos: Los pesos $\mathbf{U}_i(t)$ son modificados según la ley de aprendizaje.

$$\mathbf{U}_i(t+1) = \begin{cases} \mathbf{U}_i(t) + \alpha(t).h_{ci}(t).\left(\mathbf{x}(t) - \mathbf{U}_i(t)\right) & i \in N_c(t) \\ \mathbf{U}_i(t) & i \notin N_c(t) \end{cases} \qquad (D.16)$$

La función $h_{ci}(t)$ se conoce como función de retroalimentación del nodo i al nodo ganador c. Para la convergencia del proceso de aprendizaje, es necesario que $h_{ci}(t) \to 0$ cuando $t \to \infty$. Usualmente viene definida por la función:

$$h_{ci}(t) = \exp\left(-\frac{\left\|\mathbf{U}_c(t) - \mathbf{U}_i(t)\right\|^2}{2\sigma^2(t)}\right) \qquad (D.17)$$

Donde $\sigma(t)$ y $\alpha(t)$ son identificados como el factor radial y de aprendizaje respectivamente, $\alpha(t)$ varía entre 0 y 1, éste decrece con el número de iteraciones t del proceso de entrenamiento.

6. Modificación de las tasas de aprendizaje: Se actualiza el factor radial y de aprendizaje, a través de las relaciones:

$$\alpha(t+1) = \alpha(t).f_\alpha \qquad (D.18)$$

$$\sigma(t+1) = \sigma(t).f_\sigma \qquad (D.19)$$

Donde f_α es definida como la constante de aprendizaje, teniendo un valor cercano a 1 y f_σ definida como la constante radial que también presenta un valor cercano a 1.

7. Volver al paso 2: Mientras no se haya alcanzado un estado de "orden" en la malla de competición, el algoritmo regresa al nivel 2 hasta que los nodos estén adecuadamente afinados.

Es usual utilizar una estrategia separada en la ordenación y en la convergencia. Así durante el aprendizaje habrá una evolución tanto de la función de

vecindad como del factor de aprendizaje mientras en la convergencia se mantiene la vecindad siguiendo decreciendo.

D.7.1 El SOFM como cuantizador óptimo de vectores

El objetivo de la cuantificación vectorial consiste en tratar de reducir la cantidad de información requerida para almacenar o transmitir un conjunto numeroso de vectores de un cierto espacio multidimensional, reemplazando cada vector por otro elegido de entre un conjunto de vectores de referencias óptimo [Del Brío y Sanz, 2002, pp. 106-108]. Para que el SOFM funcione como un cuantizador óptimo de vectores, no es necesario establecer alguna relación topológica entre los vectores de referencia. Por lo tanto, un mapa auto-organizado que funciona como cuantizador óptimo de vectores constituye una memoria o tabla de búsqueda de tamaño limitado, que al no poder almacenar cantidades masivas de datos, guarda en su lugar un conjunto de patrones representativos de la entrada (los vectores de referencia) que aseguren un aprovechamiento óptimo de la memoria, preservando además en cierto modo su topología.

Anexo E
El DSP Starter Kit C6711 (DSK)

E.1. Introducción

El *DSP Starter Kit* 'C6711 es una potente plataforma DSP de *Texas Instruments* (TI) que permite al usuario desarrollar, analizar y probar aplicaciones con el procesador TMS320C6711. Este *kit* incluye una tarjeta de circuito impreso que sirve como hardware de referencia para realizar nuevos diseños, denominada Tarjeta DSK y un extenso software para crear y probar algoritmos de tiempo real, denominado *Code Composer Studio v2*.

E.2. Tarjeta DSK

La tarjeta DSK ha sido diseñada para realizar operaciones de escritorio mientras está conectada al puerto paralelo del computador, tiene dimensiones aproximadas de 5 pulgadas de ancho, 8 pulgadas de largo y 0.7 pulgadas de alto (ver figura E.1).

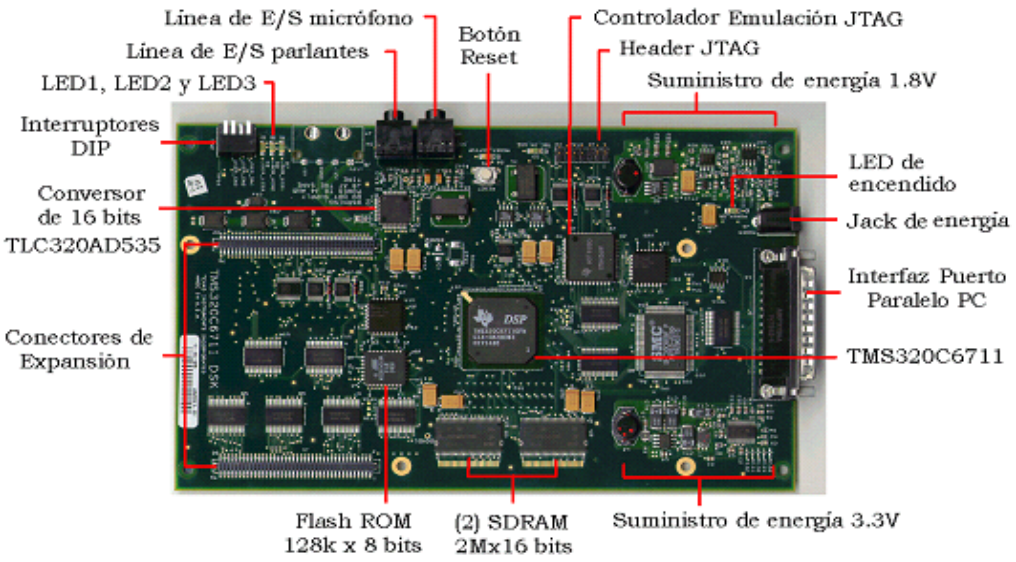


Figura E.1 Tarjeta DSK del DSP *Starter Kit* TMS320C6711.

El diagrama de bloques de las funciones básicas e interfaces de la tarjeta DSK se muestra en la figura E.2, siendo los principales componentes que posee:

(a) Un procesador TMS320C6711 que trabaja a 150 MHz, capaz de ejecutar hasta 900 millones de operaciones de punto flotante por segundo (MFLOPS).

(b) Soporte dual de reloj, 150MHz para la CPU y 100MHz para la interfaz de memoria externa (EMIF).

(c) Una interfaz controladora de puerto paralelo para comunicarse con el computador a través de los modos SPP bidireccional o EPP.

(d) Una memoria dinámica síncrona de acceso aleatorio (SDRAM) de 16 Mbytes con una frecuencia de 100 MHz.

(e) Una memoria de sólo lectura (ROM) programable y borrable flash de 128 kbytes.

(f) Un puerto de E/S de 8 bits mapeado en memoria.

(g) Emulación JTAG integrada a través del puerto paralelo y un chip XDS510 externo de soporte.

(h) Una interfaz de puerto *host* (HPI) para acceder a toda la memoria del DSP a través del puerto paralelo.

(i) Un *codec* de audio de 16 bits.

(j) Reguladores de voltaje conmutados para 1.8 V_{DC} y 3.3 V_{DC}.

(k) Seis indicadores LED (diodos emisores de luz).

(l) Expansión de memoria y conectores periféricos para soporte de tarjetas externas.

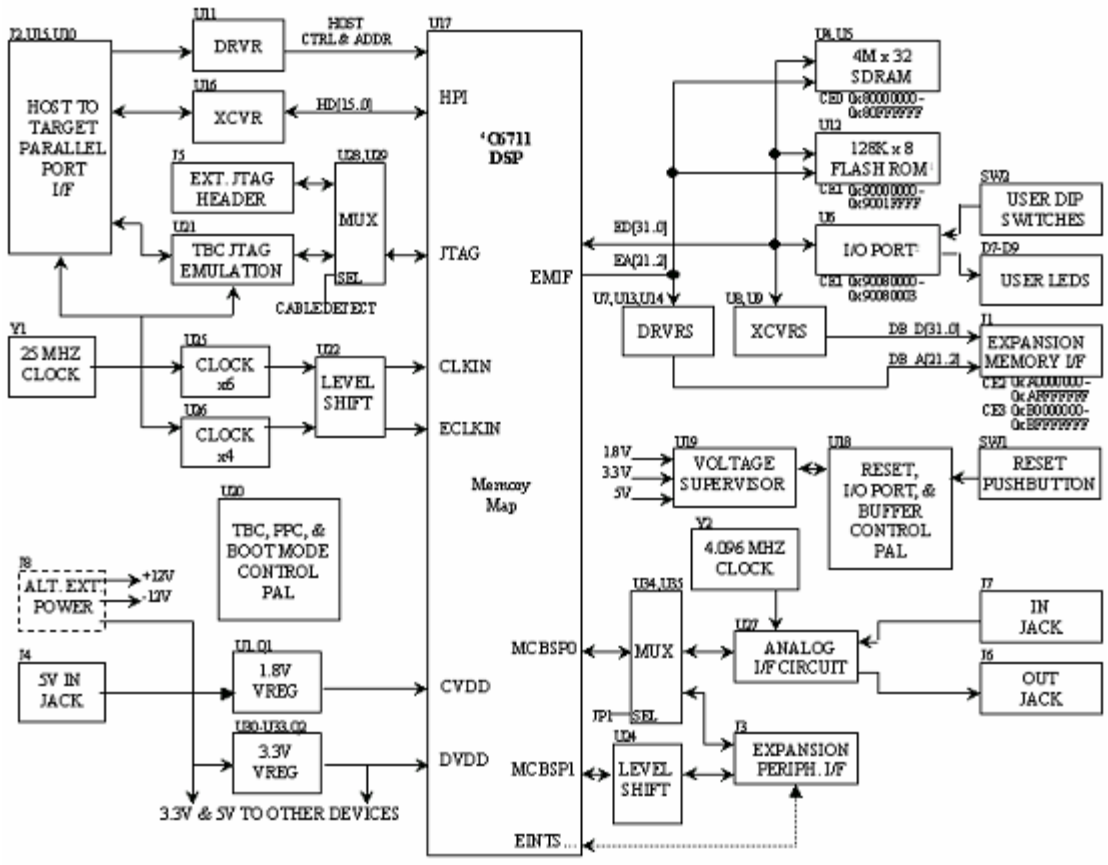


Figura E.2 Diagrama de bloques de las funciones básicas e interfaces de la tarjeta DSK.

E.2.1. El procesador TMS320C6711

Este chip pertenece a la familia C6000 de TI, consta de tres bloques fundamentales: la unidad central de procesamiento (*Central Processing Unit* – CPU), los periféricos y la memoria interna, la figura E.3 muestra el diagrama de bloques de este procesador.

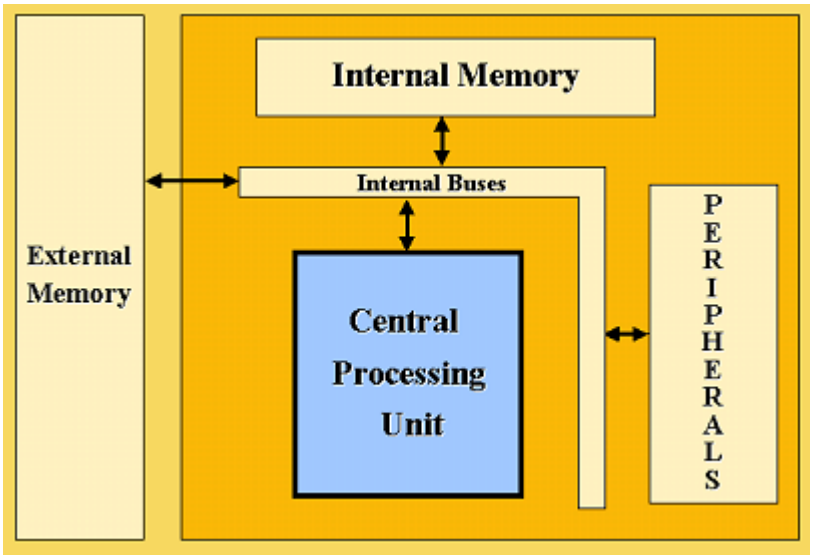


Figura E.3 Diagrama de bloques general del procesador TMS320C6711.

E.2.1.1. Arquitectura del TMS320C6711

El procesador posee espacios de memoria separados para el programa y los datos, similar a la arquitectura *Harvard*; Estos tipos de procesadores de última generación poseen una arquitectura conocida como Palabra de Instrucción Muy Larga [Lorenzi, 2000] (*Very Large Instruction Word* - VLIW) en donde varias instrucciones son capturadas y procesadas simultáneamente a través de un *Fetch Packet*. En este procesador, ocho instrucciones de 32 bits pueden ser capturadas todas a la vez desde la memoria del chip en el bus de datos de 256 bits. TI modificó la arquitectura VLIW para que cada *Fetch Packet* esté hecho de *Execute Packets* (EP). Cada EP contiene instrucciones que pueden ser ejecutadas al mismo tiempo, en paralelo (ver figura E.4). El bit menos significativo de cada instrucción de 32 bits determina si la siguiente instrucción pertenece al mismo EP o es parte del siguiente EP. Esta modificación es llamada *VelociTI*, comparada con la arquitectura VLIW, incrementa el desempeño (reduce el tamaño del código).

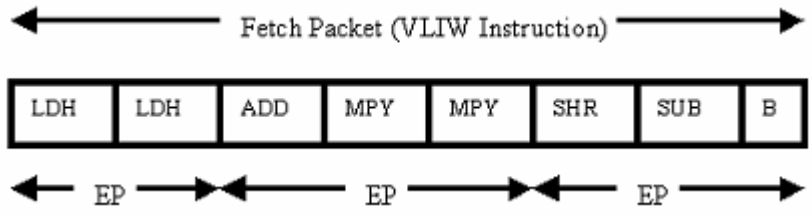


Figura E.4 Esquema de las características en la arquitectura del TMS320C6711.

E.2.1.2. Unidad central de procesamiento

La CPU contiene los siguientes componentes:

(a) Unidad *fetch* de programa.

(b) Unidad de despacho de instrucción.

(c) Unidad de decodificación de instrucción.

(d) 32 registros de 32 bits.

(e) Dos caminos de datos (*path*), cada uno con cuatro unidades funcionales.

(f) Registros de control.

(g) Lógica de control.

(h) Lógica de interrupción, emulación y prueba.

Como se muestra en la figura E.5, la CPU presenta dos caminos de datos (A y B), cada camino tiene cuatro unidades funcionales (.L, .S, .M y .D) y una colección de 16 registros de 32 bits (*register file*). Las unidades funcionales ejecutan operaciones lógicas, de desplazamiento, multiplicación y direccionamiento de datos. Todas las instrucciones aceptan operaciones de carga y almacenamiento sobre los registros. Las dos unidades de direccionamiento de datos (.D1 y .D2) son exclusivamente responsables de toda la transferencia de datos entre las colecciones de registros y la memoria [Texas Instruments, 2001a].

E.2.1.3. Caminos de datos de la CPU

Los caminos de datos de la CPU consisten de:

(a) Dos colecciones de registros de propósito general (A y B).

(b) Ocho unidades funcionales (.L1, .L2, .S1, .S2, .M1, .M2, .D1 y .D2).

(c) Dos caminos de lectura de memoria (LD1 y LD2).

(d) Dos caminos de almacenamiento en memoria (ST1 y ST2).

(e) Dos caminos cruzados entre las colecciones de registros (1X y 2X).

(f) Dos caminos de direccionamiento de datos (DA1 y DA2).

La figura E.6 muestra el camino de los datos de la CPU.

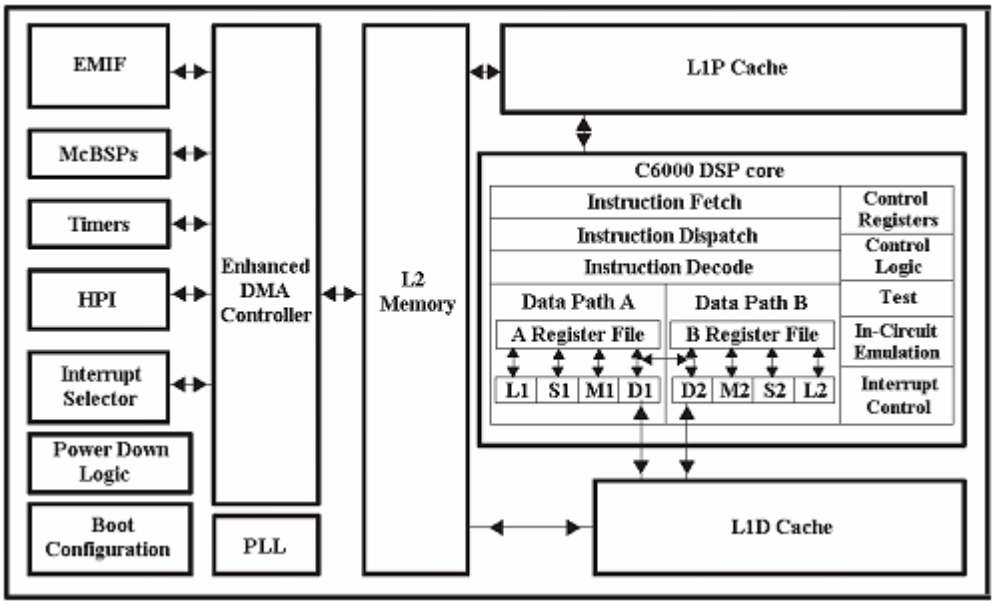


Figura E.5 Diagrama de bloques específico del procesador TMS320C6711.

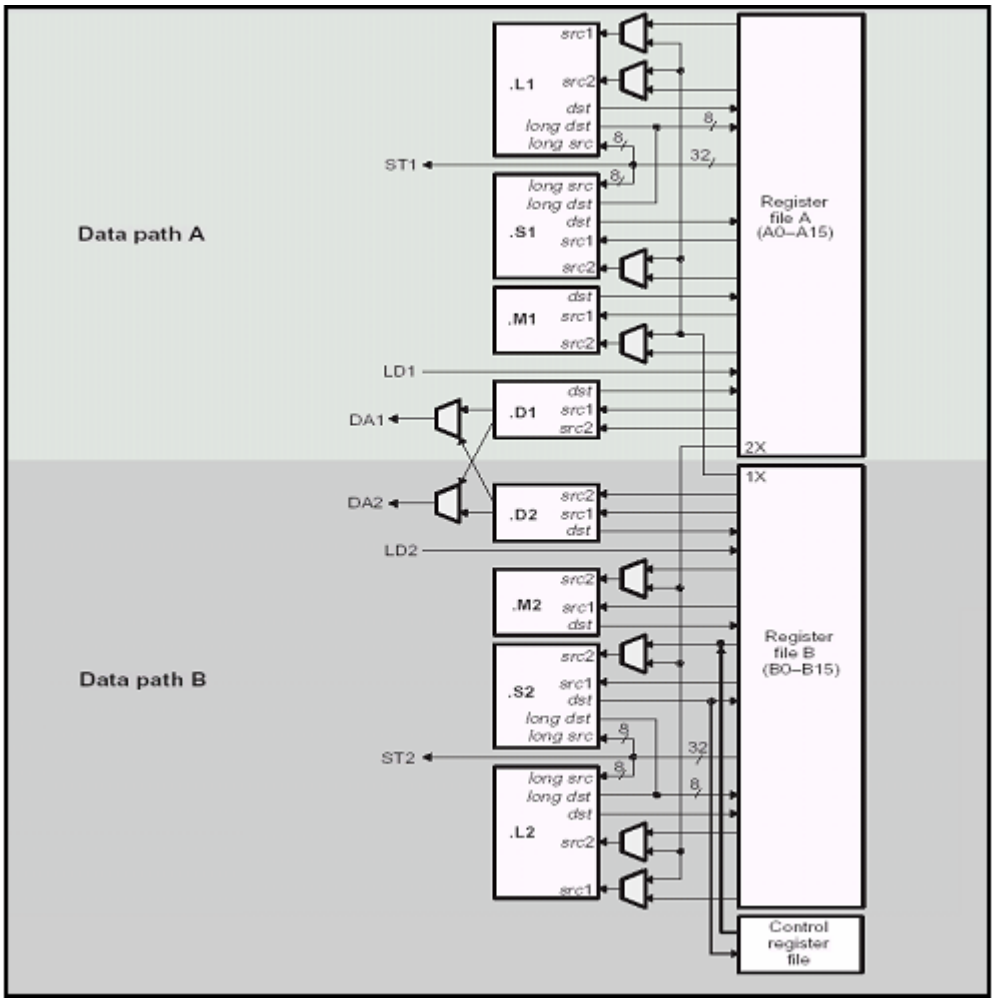


Figura E.6 Caminos de datos de la CPU del TMS320C6711.

E.2.1.4. Colección de registros de propósito general

Hay dos colecciones de registros de propósito general (A y B) en los caminos de datos de la CPU. Cada una de esas colecciones contiene 16 registros de 32 bits (A0-A15) para la colección A y (B0-B15) para la colección B. Los registros de propósito general pueden utilizarse para manejar datos o punteros de direccionamiento. Los registros A1, A2, B0, B1 y B2 pueden ser utilizados como registros de condición. Los registros A4-A7 y B4-B7 pueden ser usados para el direccionamiento circular.

Las colecciones de registros de propósito general soportan datos de 32 y 40 bits de punto fijo. Los datos de 32 bits, pueden estar contenidos en cualquier registro de propósito general. Los datos de 40 bits están contenidos en dos registros; los 32 bits menos significativos del dato son colocados en un registro par y los restantes ocho bits más significativos son colocados en los ocho bits menos significativos del registro próximo superior (que es siempre un registro impar). La CPU también usa ese par de registros para colocar valores de punto flotante de doble precisión de 64 bits.

E.2.1.5. Unidades funcionales

Las ocho unidades funcionales en los caminos de datos de la CPU pueden ser divididas en dos grupos de cuatro; cada unidad funcional en un camino de datos es casi idéntica a la unidad correspondiente en el otro camino de datos. Las unidades funcionales se describen en la Tabla E.1. La mayoría de los caminos en la CPU, soportan operaciones de 32 bits, y algunas soportan operaciones largas (40 bits). Cada unidad funcional tiene su propio puerto de escritura de 32 bits, en una colección de registros de propósito general. Todas las unidades terminan en 1 (por ejemplo, .L1) cuando se refiere a la colección de registros A y en 2 cuando se refiere a la colección de registros B. Cada unidad funcional tiene dos puertos de lectura de 32 bits, para cada operando src1 y src2. Cuatro unidades (.L1, .L2, .S1 y .S2) tienen un puerto extra de 8 bits para ejecutar operaciones de 40 bits. Debido a que cada unidad tiene su propio puerto de escritura de 32 bits, las ocho unidades pueden ser usadas en paralelo en cada ciclo.

E.2.1.6. Colección de registros de control

Una unidad (.S2) puede leer de y escribir hacia los registros de control. La Tabla E.2, menciona y describe, los registros contenidos en la colección de registros de control. El TMS320C6711 posee tres registros de configuración adicionales, para soportar operaciones de punto flotante (vea la Tabla E.3). Los registros especifican los modos de redondeo de punto flotante para las unidades .M y .L. También contienen campos de bits para advertir si src1 y src2 no son números (*Not*

a Number – NaN) o números no normalizados, además si resulta un *overflow* o un *underflow,* es inexacto, es infinito o no válido. Hay campos que advierten si una división por cero fue ejecutada o si una comparación fue ejecutada con un NaN.

Unidad Funcional	Operaciones en punto fijo	Operaciones en punto flotante
Unidad .L(.L1, .L2)	Operaciones aritméticas y comparación de 32 y 40 bits. Cuenta de 0's y 1's mas a la izquierda, para 32 bits. Normalización de 32 y 40 bits. Operaciones lógicas de 32 bits	Operaciones aritméticas. Operaciones de conversión: DP->SP, INT->DP, INT->SP
Unidad .S(.S1, .S2)	Operaciones aritméticas de 32 bits. Corrimientos de 32/40 bits, y operaciones campos de bits en 32 bits. Saltos. Generación de constantes. Transferencia de registros de/hacia registros (solamente .S2)	Comparación recíproca. Operaciones de raíz cuadrada. Operaciones de valor absoluto. Operaciones de conversión SP a DP
Unidad .M(.M1, .M2)	Operaciones de multiplicación de 16x16 bits	Operaciones de multiplicación de 32x32 bits. Operaciones de multiplicación de punto flotante
Unidad .D(.D1, .D2)	Sumas, restas y cálculos de direccionamiento circular de 32 bits. Carga y almacenamiento con offset constante de 5 bits. Carga y almacenamiento con offset constante de 15 bits (sólo .D2)	Lectura de palabras dobles con offset constante de 5 bits

Tabla E.1 Unidades funcionales y las operaciones realizadas.

Abreviatura	Nombre	Descripción
AMR	Registro del modo de direccionamiento	Especifica si utiliza direccionamiento lineal o circular para cada uno de los ocho registros; también contiene el tamaño para el direccionamiento circular
CSR	Registro del control de estado	Contiene el bit de interrupción global, los bits de control del caché y otros bits de control de estado misceláneos
IFR	Registro de bandera de interrupción	Despliega el estado de las interrupciones
ISR	Registros para activar interrupción	Permite activar interrupciones manualmente
ICR	Registro para interrupción	Permite limpiar interrupciones pendientes manualmente
IER	Registro para retorno de interrupción	Permite habilitar/deshabilitar interrupciones individuales
NRP	Puntero de retorno de interrupción no enmascarable	Contiene la dirección de retorno de una interrupción no enmascarable
PCE1	Contador del programa, fase E1	Contiene la dirección del *packet fetch* (contiene el paquete de ejecución del pipeline) en la Etapa E1

Tabla E.2 Registros de control.

Abreviatura	Nombre	Descripción
FADCR	Registro de configuración del sumador de punto flotante	Especifica el modo underflow, modo de redondeo, NaN y otras excepciones para la unidad .L
FADUCR	Registro de configuración auxiliar de punto flotante	Especifica los modos de underflow, modos de redondeo, NaN y otras excepciones para la unidad .S
FMCR	Registro de configuración del multiplicador de punto flotante	Especifica los modos de underflow, modos de redondeo, NaN y otras excepciones para la unidad .M

Tabla E.3 Registros de control extendidos.

E.2.1.7. Caminos entre colecciones de registros (*Register File Cross Paths*)

Cada unidad funcional lee directamente de y escribe directamente hacia la colección de registros, dentro de su propio camino de datos. Esto es, las unidades .L1, .S1, .D1 y .M1 escriben en la colección de registros A y las unidades .L2, .S2, .D2 y .M2 escriben en la colección de registros B. Las colecciones de registros están conectadas a las unidades funcionales de la colección de registros opuesto, a través de los caminos cruzados 1X y 2X. Esos caminos cruzados permiten a las unidades funcionales de un camino de datos, acceder a operandos de 32 bits del lado opuesto. El camino cruzado 1X permite a las unidades funcionales del camino de datos A, leer su operando fuente desde la colección de registros B. El camino cruzado 2X permite a las unidades funcionales del camino de datos B, leer su operando fuente desde la colección de registros A.

E.2.1.8. Caminos de memoria, cargas y almacenamiento

Hay dos caminos de 32 bits para leer los datos de memoria en los registros de almacenamiento: LD1 para la colección de registros A y LD2 para la colección de registros B. El TMS320C6711 también tiene un segundo camino de carga de 32 bits para ambas colecciones de registros A y B. Este segundo camino permite a la instrucción LDDW leer simultáneamente dos registros de 32 bits en los lados A y B. Existen además dos caminos de 32 bits, ST1 y ST2, para almacenar valores de los registros a la memoria para cada colección de registros. Los caminos de lectura largos .L y .S son compartidos con los caminos de almacenamiento.

E.2.1.9. Caminos de direccionamiento de datos

Los caminos de direccionamiento de datos (DA1 y DA2) mostrados en la figura E.6 colocados fuera de las unidades .D, permiten generar direcciones de datos de una colección de registros. Con eso se sostienen cargas y almacenamientos en memoria, desde la otra colección de registros. Sin embargo, las cargas y almacenamientos ejecutados en paralelo deben cargar hacia y desde la misma colección de registros. Aunque también existe la alternativa de que ambos usen un camino cruzado a la colección opuesta.

E.2.1.10. Mapa de Memoria

El mapa de memoria general del procesador TMS320C6711 es mostrado en la figura E.7. El rango total de direcciones de memoria es de 4 GBytes (correspondientes a la representación de direcciones internas de 32 bits). El mapa de memoria está dividido en memoria interna de programa, memoria interna de datos, espacios de memoria externos y un espacio interno de periféricos.

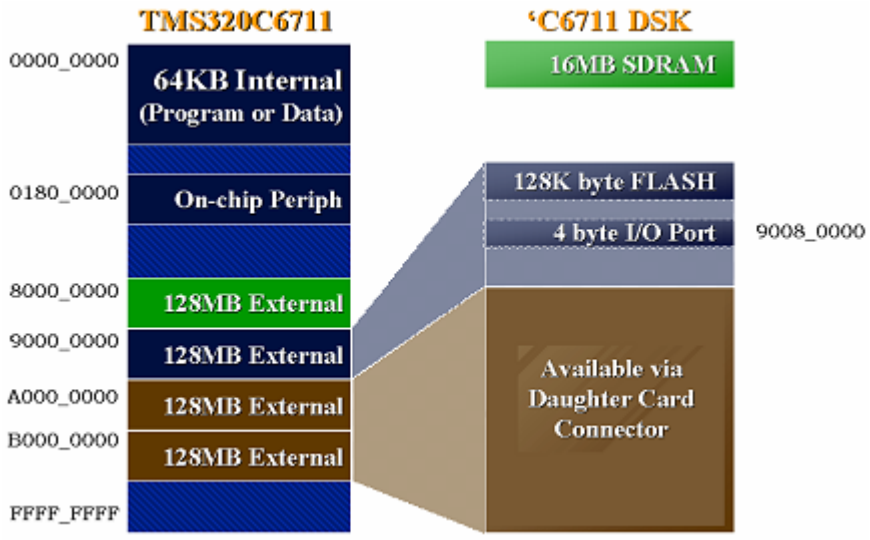


Figura E.7 Distribución global de la memoria en el TMS320C6711.

E.2.1.11. Modos de direccionamiento

Los modos de direccionamiento por defecto son lineales aunque también existe el modo de direccionamiento circular. El modo de direccionamiento se especifica con el registro modo de direccionamiento (AMR). Con todos los registros se puede ejecutar el direccionamiento lineal. Sólo en ocho de ellos se puede ejecutar el direccionamiento circular: del A4 a A7 (que son usados por la unidad .D1) y del B4 a B7 (que son usados por la unidad D2). Ninguna otra unidad puede ejecutar direccionamiento circular. La CPU posee una arquitectura de carga/almacenamiento, lo que significa que la única manera de acceder datos en memoria es con la instrucción de carga o almacenamiento.

E.2.1.12. Periféricos

Los periféricos que son accesibles al usuario se configuran con un conjunto de registros de control mapeados en memoria. El controlador del bus de periféricos realiza el arbitraje para el acceso a los periféricos. La lógica de configuración de *Boot* está conectada por señales externas y la lógica de baja energía es accesible directamente por la CPU [Texas Instruments, 2001b].

Controlador DMA: Transfiere datos entre rangos de direccionamiento en el mapa de memoria sin intervención de la CPU. El controlador DMA tiene cuatro canales programables y cinco canales auxiliares.

Controlador EDMA: El controlador DMA mejorado realiza las mismas funciones del controlador DMA. El EDMA tiene dieciséis canales programables, así como un espacio de RAM para soportar múltiples configuraciones de futuras transferencias.

HPI: Es un puerto de comunicación paralela por medio del cual, un procesador *host* puede acceder directamente al espacio en memoria de la CPU. El dispositivo *host* tiene facilidad de acceso debido a que es el maestro de la interfaz. El *host* y la CPU pueden intercambiar información a través de la memoria interna o externa. En suma, el *host* tiene acceso directo a los periféricos de la memoria mapeada.

Bus de Expansión: Es un reemplazo para el HPI, así como una expansión del EMIF. La expansión proporciona dos áreas distintas de funcionalidad, (puerto *host* y puertos de Entrada/Salida) que pueden coexistir en un sistema. El puerto *host* del bus de expansión puede operar en modo asíncrono (esclavo), similar al HPI o en modo síncrono (maestro/esclavo). Estos modos permiten la interfaz entre dispositivos para una variedad de protocolos del *host*. FIFOs síncronos y los dispositivos periféricos de Entrada/Salida asíncronos pueden conectarse al bus de expansión.

EMIF: Soporta una interfaz de baja adherencia (*glueless*) para varios dispositivos externos, que incluyen:

(a) SRAM síncrona (SBSRAM).

(b) DRAM síncrona (SDRAM).

(c) Dispositivos asíncronos, incluyendo SRAM, ROM y FIFOs.

(d) Dispositivo externo de memoria compartida.

Configuración del *Boot*: Determinan las acciones de inicialización que ejecuta el DSP después de activado el reset del dispositivo. Estas incluyen: cargas de código de un espacio externo de ROM sobre el EMIF y cargas de código a través del HPI/bus de un host externo.

McBSP: El puerto serial multicanal con *buffer* está basado en las interfaces

estándar del puerto serie. El puerto puede almacenar muestras seriales en un *buffer* de memoria automáticamente, con la ayuda del controlador DMA/EDMA. Este también tiene capacidad de multicanal, compatible con algunos estándares de conexión de redes. Proporciona:

(a) Comunicación *full-duplex*.

(b) Registros de datos de doble buffer para flujo continuo de datos.

(c) Tramado independiente y temporizado para dispositivos y transmisión.

(d) Interfaz directa a *codecs* estándar, *chips* de interfaz analógica (AICs) y otros dispositivos A/D y D/A conectados en forma serial.

Tiene las siguientes capacidades:

(a) Transmisión y recepción multicanal de 128 canales.

(b) Un selector del ancho del tamaño del dato, que incluye 8, 12, 16, 20, 24 y 32 bits.

(c) Transferencia inicial de 8 bits con LSB o MSB.

(d) Polaridad programable para ambas tramas de sincronización y relojes de datos.

(e) Reloj interno altamente programable y generación de trama.

TIMER: Tiene dos *timers* de propósito general que son usados para:

(a) Eventos del *timer.*

(b) Eventos de contador.

(c) Generador de pulsos.

(d) Interrupción de la CPU.

(e) Enviar eventos de sincronización al controlador DMA/EDMA.

E.3. Code Composer Studio (CCS)

Como se describió anteriormente, el procesador TMS320C6711 es muy poderoso por sí mismo, pero para desarrollar programas se requiere una arquitectura de soporte software para almacenar programas, datos y enviar o recibir señales con la tarjeta DSK.

El CCS es un potente entorno de desarrollo que provee una útil transición entre un programa DSP de alto nivel y un programa de lenguaje máquina en una

tarjeta impresa. El CCS consiste de un conjunto de herramientas de software y librerías para desarrollar programas DSP, compilarlos en código máquina y escribirlos en la memoria este chip y en la memoria externa de la tarjeta DSK. Además contiene herramientas de diagnóstico para analizar, depurar y rastrear cómo los algoritmos se han implementado [Texas Instruments, 2001c]. En este trabajo se utilizó el CCS para desarrollar y compilar programas que luego fueron descargados del computador hacia el hardware del procesador DSP.

El CCS incluye los siguientes componentes: (ver la figura E.8)
Herramientas de generación de código para TMS320C6000
Entorno en Desarrollo integrado (IDE)
DSP/BIOS plug-ins y API's
RTDX plug-ins, interfaz host y API's

E.3.1. Herramientas de desarrollo para la generación de código

Los TMS320C6x soportan un conjunto de herramientas, para el desarrollo del software, que incluyen: compilador C/C++ optimizado, ensamblador optimizado, ensamblador, enlazador y las herramientas asociadas. Además, los TMS320C6x soportan las siguientes herramientas, para el desarrollo del lenguaje ensamblador.

Ensamblador (*Assembler*)
Archivador (*Archiver*)
Enlazador (*Linker*)
Listado Absoluto (*Absolute lister*)
Listado de referencias cruzadas (*Cross-reference lister*)
Utilidad de conversión hexadecimal (*Hex conversion utility*)

La figura E.9 muestra el flujo para el desarrollo de software con los TMS320C6x. La parte sombreada, representa el camino más común de desarrollo; las demás partes son opcionales.

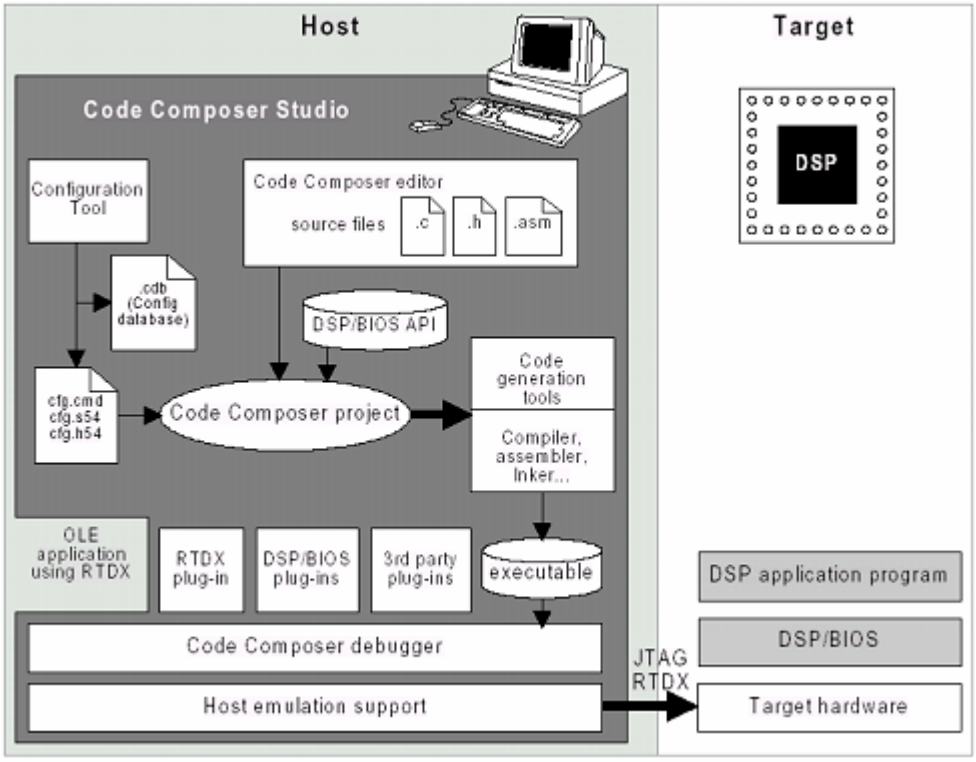


Figura E.8 Principales herramientas del *Code Composer Studio* ‘C6000.

E.3.2. Descripción de las Herramientas del CCS

El optimizador de ensamble (*assembly optimizer*): Permite escribir código en lenguaje ensamblador lineal, sin importar la estructura del *pipeline* o la asignación de registros. Asigna registros y usa optimización de ciclos, para convertir el código, de lenguaje ensamblador lineal a lenguaje ensamblador en paralelo.

Compilador C/C ++ (C/C++ *compiler*): Acepta el código fuente en lenguaje C/C++ y produce código fuente en lenguaje ensamblador para los TMS320C6x.

El ensamblador (*assembler*): Traduce los archivos de lenguaje ensamblador fuente en lenguaje de máquina.

El enlazador (*linker*): Combina los archivos objeto a un sólo archivo ejecutable para el DSP. También acepta archivos de librerías y módulos de salidas,

creados por una ejecución previa del mismo.

El archivador (*archiver*): Permite juntar un grupo de archivos dentro de un sólo archivo, llamado librería. Por ejemplo, se puede juntar distintas macros, dentro de una librería de macros.

También se puede usar la utilería de construcción de librerías (*library-build utility*) para construir una librería propia de soporte en tiempo real.

El Listado absoluto (*absolute lister*): Es una herramienta para depuración de programas. Acepta como entrada, archivos objetos enlazados y crea archivos con extensión ABS que son ensamblados para producir una lista que muestra las direcciones absolutas de código objeto.

La utilidad de conversión hexadecimal (*hex conversion utility*): Convierte un archivo objeto de tipo COFF en un archivo cuyo formato objeto se puede seleccionar entre los siguientes: TI-Tagged, ASCII hexadecimal, Intel, Motorola-S o Tektronix. El archivo convertido, puede ser transmitido a una memoria EPROM.

El listado de referencias cruzadas (*Cross-reference lister*): Utiliza archivos objeto para producir un listado de referencias que muestra símbolos, definiciones y sus referencias en el archivo fuente enlazado.

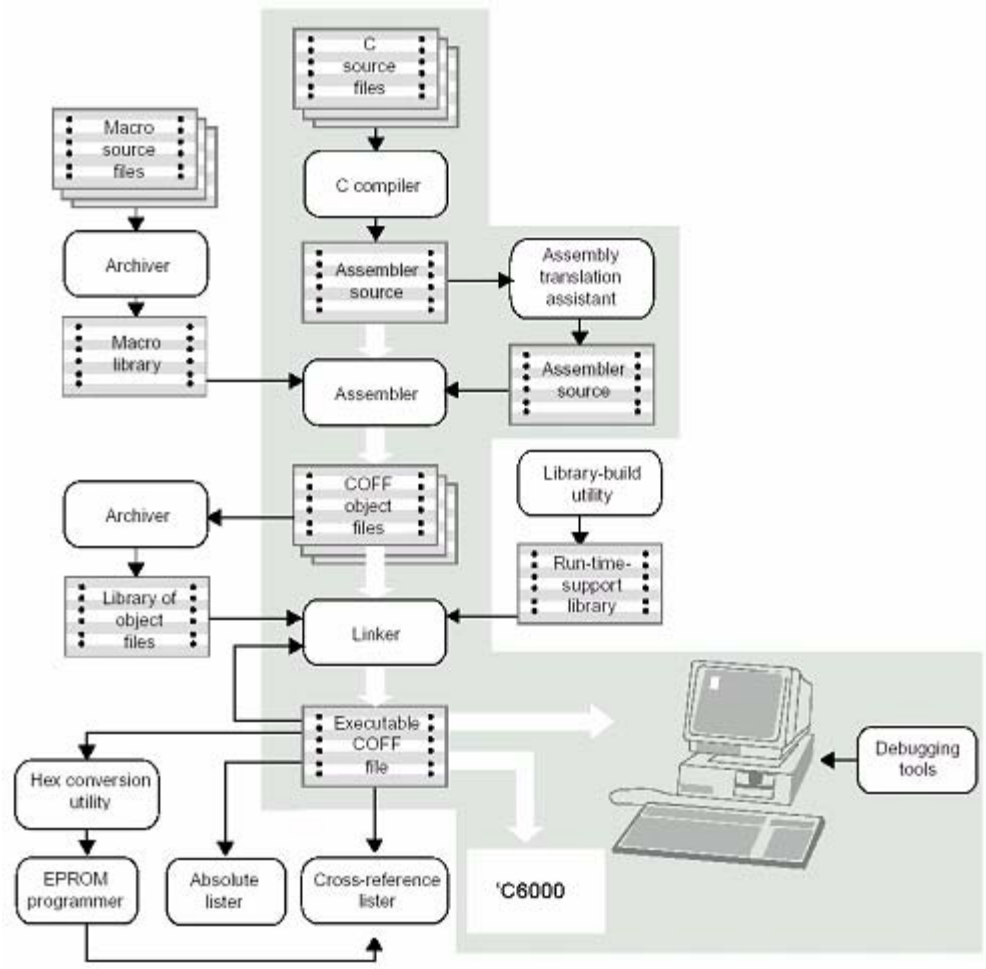


Figura E.9 Flujo para el desarrollo de programas con el TMS320C6711.

E.3.3. Entorno de desarrollo integrado

El IDE del CCS está diseñado para permitirle editar, construir y depurar sus programas ejecutados por el DSP. Dentro del CCS se crea una aplicación añadiendo archivos a un proyecto. El archivo de proyecto es utilizado para construir la aplicación. Los archivos en un proyecto pueden incluir archivos fuente en C y *assembler*, archivos objeto, librerías, archivos de comandos del enlazador y archivos de cabecera. Puede utilizar la ventana de proyecto para especificar las opciones que desea usar cuando compila, ensambla y enlaza un proyecto.

El depurador del CCS le permite realizar las siguientes actividades:

(a) Establecer puntos de ruptura (*breakpoints*) con varias opciones de ejecución por pasos.

(b) Actualizar automáticamente ventanas en puntos de ruptura.

(c) Visualizar variables de programa.

(d) Visualizar y editar la memoria y los registros.

(e) Visualizar las llamadas a la pila (*stack*).

(f) Usar puntas de prueba para flujo de datos.

(g) Graficar señales que están siendo procesadas.

(h) Ver y ejecutar instrucciones en C desensambladas.

Además el CCS también provee el lenguaje GEL, que le permite a los desarrolladores añadir funciones a los menús del CCS para tareas comúnmente realizadas.

E.3.4. DSP/BIOS

Durante el análisis del ciclo de desarrollo del software, las características tradicionales de depuración no son efectivas para diagnosticar problemas que nacen de bucles dependientes del tiempo. El CCS provisto del DSP/BIOS le permite realizar tareas de análisis en tiempo real, permitiéndole monitorear una aplicación DSP con un mínimo impacto en el desempeño en tiempo real [Texas Instruments, 2001d]. El DSP/BIOS también permite crear archivos de configuración que define objetos utilizados por la Interfaz de Programación de Aplicaciones (*Applications Programming Interface* - API).

El API del DSP/BIOS provee de las siguientes capacidades de análisis en tiempo real:

(a) Muestra los eventos y el flujo de control dinámico durante la ejecución del programa.

(b) Monitorear el desempeño y los recursos del DSP.

(c) Enlazar objetos de entrada y salida del DSP a archivos de datos en el computador.

E.3.5. RTDX

Los DSPs de TI proveen de un soporte de emulación en el chip que le permite al CCS controlar la ejecución de un programa y monitorear en tiempo real la actividad del programa. La comunicación se realiza a través de un enlace JTAG mejorado. Este enlace es una forma de conectarse a cualquier sistema DSP sin afectar de manera considerable el desempeño de la tarea que está siendo realizada por el procesador DSP. Permitiéndole una comunicación bidireccional en tiempo real entre el DSP y el computador (ver figura E.10).

RTDX provee visibilidad continua de la forma cómo los DSP operan en el mundo real. Además RTDX permite a los desarrolladores transferir datos entre el computador y el DSP sin detener la aplicación que está siendo ejecutada por el DSP. Los datos pueden ser analizados y visualizados en el computador usando cualquier cliente de automatización OLE. Esto reduce el tiempo de desarrollo dándole a los diseñadores una representación real de la manera en que su sistema verdaderamente trabaja.

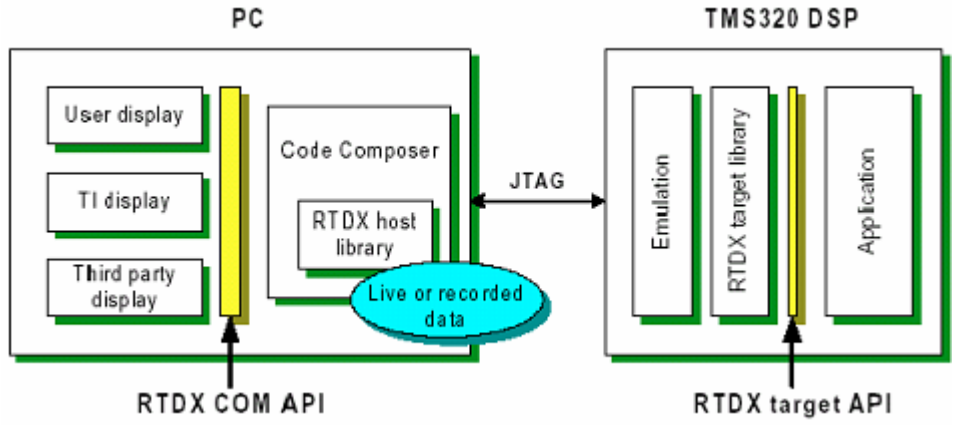


Figura E.10 Esquema general del RTDX.

En la plataforma del computador (*host*) opera una librería RTDX junto con el CCS. Herramientas de análisis y visualización comunican al RTDX a través de una API COM para recibir o enviar los datos a la aplicación del DSP. Los diseñadores también pueden utilizar paquetes estándares de software tales como LabVIEW, Excel, Visual Basic, Visual C++, entre otros.

Referencias Bibliográficas

[Aching y Rojas, 2003] ACHING, J.; ROJAS, D. *"Algoritmos para el Reconocimiento de Imágenes de Huellas Dactilares"*. Revista Electrónica - UNMSM, No. 12, pp. 11-20, 2003.

[Acton et al., 2001] ACTON S., PRASAD D., HAVLICEK J., CONRAD A. "*Oriented Texture Completion by AM-FM Reaction-Difussion*". IEEE Transactions on Image Processsing, vol. 10, No. 6, pp. 885-896, 2001.

[Adler y Rao, 1996] ADLER, J.; RAO, B. *"SurfWave: an object-oriented class library for wavelet analysis"*. IEEE International Conference on Acoustics, Speech, and Signal Processing, vol. 3, pp. 1451-1454, 1996.

[Bazen y Gerez, 2000] BAZEN, A.; GEREZ, S. *"Directional field computation for fingerprints based on the principal component analysis of local gradients"*. 11th Annual Workshop on Circuts, Systems and Signal Processing, Veldhoven, Netherlands, 2000.

[Bazen et al., 2000] BAZEN, A.; GEREZ, S.; VERWAAIJEN, G. "*A Correlation-Based Fingerprint Verification System*". 11th Annual Workshop on Circuts, Systems and Signal Processing, Veldhoven, Netherlands, 2000.

[Bazen y Gerez, 2001] BAZEN, A; GEREZ, S. *"Segmentation of fingerprint images"*. In Proc. ProRISC2001, 12th Annual Wokshop on Circuits, Systems and Signal Processing, Veldhoven, Netherlands, 2001.

[Bazen et al., 2002] BAZEN, A; KLEIN, S.; VELDHUIS, R. *"Fingerprint Image Segmentation Based on Hidden Markov Models"*. In Proceedings ProRISC, pp. 310-318, 2002.

[Bigün et al., 1991] BIGÜN, J.; GRANLUND, G.; WIKLUND, J.. *"Multidimensional orientation estimation with applications to texture analysis and optical flow"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 775–790, 1991.

[Brown y Shvaytser, 1990] BROWN, L.; SHVAYTSER, H.. *"Surface orientation from projective foreshortening of isotropic texture autocorrelation"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, pp. 584–588, 1990.

[Bobung, 2002] BOBUNG, S. "*Evaluation of Transformation Methods to Detect Structures in Fingerprints*". Bachelor thesis. Technical University of Hamburg-Harburg, 2002.

[Capelli et al., 2000] CAPELLI, R.; MAIO, D.; MALTONI, D. *"Indexing Fingerprint Databases for Efficient 1:N Matching"*. In Proceedings Int. Conference on Control Automation Robotics and Vision (6th), 2000.

[Ceguerra y Koprinska, 2002] CEGUERRA, A.; KOPRINSKA, I. *"Integrating Local and Global Features in Automatic Fingerprint Verification"*. In Proceedings Int. Conference Pattern Recognition (16th), vol. 3, pp. 347–350, 2002.

[Coetzee y Botha, 1993] COETZEE, L.; BOTHA, E. *"Fingerprint Recognition in Low Quality Images"*. Pattern Recognition, vol. 26, No. 10, pp. 1441-1460, 1993.

[Cortijo, 1995] CORTIJO, F. *"Estudio Comparativo de Métodos de Clasificación de Imágenes Multibanda"*. Tesis Doctoral, Universidad de Granada, Departamento de Ciencias de la Computación e Inteligencia Artificial, España, 1997.

[Crouzil et al., 1996] CROUZIL, A.; MASSIP-PAILHES, L. CASTAN, S.

"A New Correlation Criterion Based on Gradient Fileds Similarity". In Proceedings Int Conference on Pattern Recognition (13th), pp 632-636, 1996.

[Chang et al., 1997] CHANG, S.; CHENG, F.; HSU, W.; WU, G. *"Fast Algorithm for Point Pattern-Matching"*. Pattern Recognition, vol. 30, No 2, pp. 311-320, 1997.

[Chen et al., 2002] CHEN P.; MOON Y.; CAO K.; CHAN T. "*Enabling Fingerprint Authentication in Embedded Systems for Wireless Applications*". Departament of Computer Science and Engineering, The Chinese University of Hong Kong, 2002.

[Daugman, 1993] DAUGMAN, J. "*High confidence recognition of persons by a test of statistical independence*". IEEE trans. Pattern Anal, and Machine Intelligence, vol 16, No. 11, pp. 1148-1161. 1993.

[Domeniconi, 1997] DOMENICONI, C. *"Direct gray scale ridge reconstruction in fingerprint images"*. PhD Thesis, International Institute for Advanced Scientific Stidies, Italy, Salerno,1997.

[De Oliveira, 2000] DE OLIVEIRA, A. *"Técnicas Computacionais para Dinámica dos Fluidos"*. Editora da Universidad de São Paulo, 2000.

[De la Escalera, 2001] DE LA ESCALERA, A. *"Visión por Computador"*. Prentice Hall, Primera edición, 2001.

[Del Brio et al., 1995] DEL BRIO M.; MEDRANO, B.; RAMIREZ, I.; DOMINGUEZ, J.; BARQUILLAS, J.; BLASCO, J.; GARCIA, J. "*Short-term electric power load-forecasting using neural networks. Part I: Self-organizing networks for classification of day-types*". 145th IASTED Int.conference on Modelling, Identif. and Control, Australia, Febrero, 1995.

[Del Brio y Sanz, 2002] DEL BRIO M.; SANZ A. "*Redes Neuronales y*

Sistemas difusos 2da. Edición". Rama, Madrid, 2002.

[Duda et al., 2001] DUDA, R.; HART, P.; STORK, D. *"Pattern Classification"*. Jhon Willey & Sons, inc. USA, 2001.

[Embree y Kimball, 1991] EMBREE P., KIMBALL B., "*C Language Algorithms for Digital Signal Processing*". Prentice Hall, 1991.

[Föstner y Gülch, 1987] FÖSTNER, M.; GÜLCH, E.. *"A fast operator for detection and precise location of distinct points, corners and centers of circular features"*. ISPRS Intercommission Workshop, Interlaken, Switzerland, 1987.

[Farina et al., 1999] FARINA, A.; KOVÁCS-VAJNA, Z.; LEONE A. *"Fingerprint minutiae extraction from skeletonized binary images"*. Pattern Recognition, pp. 877-889, 1999.

[Freeman y Skapura, 1993] FREEMAN J.; SKAPURA D. "*Redes neuronales. Algoritmos, aplicaciones y técnicas de programación*". Editorial Addison-Wesley/Díaz de Santos, 1993.

[Fukuda y Shibata, 1992] FUKUDA, T.; SHIBATA, T. *"Theory and applications of Neural Networks for Industrial Control Systems"*. IEEE Transactions on Industrial Electronics, Diciembre, 1992.

[Garding y Lindeberg, 1996] GARDING, J.; LINDEBERG, T. *"Direct computation of shape cues using scale adapted spatial derivative operators"*. International Journal of Computer Vision, pp. 163–191, 1996.

[Gonzáles y Woods, 1996] GONZÁLES, R.; WOODS, R. *"Tratamiento digital de imágenes"*. Addison-Wesley/Diaz de Santos, Primera edición en Español, 1996.

[Golfarelli et al., 1997] GOLFARELLI, M.; MAIO, D.; MALTONI, D.

"*On the Error-Reject Tradeoff in Biometric Verification Systems*". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 19, No. 7, pp. 786-796, 1997.

[Guang y Milanfar, 2002] GUANG, X.; MILANFAR, P. "*Multiscale Principal Components Análisis for image local orientation estimation*". In Proceedings of the 36 Asilomar Conference on Signals Systems ans Computers, pp. 478-482, 2002.

[Hilera y Martínez, 1995] HILERA J., MARTÍNEZ V. "*Redes Neuronales Artificiales. Fundamentos, modelos y aplicaciones*". Editorial Ra-ma, 1995.

[Hong et al., 1998] HONG, L.; WAN, Y.; JAIN, A. *"Fingerprint Image Enhacement: Algorithms and Performance Evaluation"*. IEEE Transactions on Pattern Analysis and Machine intelligence, vol. 20, No. 8, pp. 777-789, 1998.

[Huvanandana et al., 2000] HUVANANDANA, S.; KIM, C.; HWANG, J.N. *"Reliable and Fast Fingerprint Identification for Security Applications"*. In Proc Conf, on Image Processing, vol. 2 pp 503-506, 2000.

[Jain et al., 1997a] JAIN, A.; HONG, L.; PANKANTI, S.; BOLLE, R. *"An Identity-Authentication System using Fingerprints"*. Proceedings of the IEEE, vol. 85, No. 9, pp. 1365-1388, 1997.

[Jain et al., 1997b] JAIN, A. ; HONG, L.; BOLLE, R. *"On – Line Fingerprint Verification"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol.19, No. 4, pp. 302-313, 1997.

[Jain et al., 1999a] JAIN, A.; PRABHAKAR; S.; HONG, L. *"A Multichannel Approach to Fingerprint Classification"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 21,

No. 4, pp. 348-359, 1999.

[Jain et al., 1999b] JAIN, S.; PRABHAKAR, L.;HONG, L. *"A Multichannel Approach to Fingerprint Classiffication"*. IEEE Trans. on Pattern Analysis And Machine Intelligence, vol 21, No. 4, pp. 348-359, 1999.

[Jain et al., 1999c] JAIN, S.; PRABHAKAR, L.; HONG, L. *"FingerCode: A Filterbank for Fingerprint Representation and Matching"*. Proceedings of IEEE Computer Society Conf. On Computer Vision and Pattern Recognition, Fort Collins, 1999.

[Jain et al., 2000] JAIN, A.K.; PRABHAKAR; S.; HONG, L. ; PANKANTI, S. *"Filterbank-Based Fingerprint Matching"*. IEEE Transactions on Image Processing, vol. 9, No. 9, pp. 846-859, 2000.

[Jain et al., 2004] JAIN A., ROSS A., PRABHAKAR S. "*An Introduction to Biometric Recognition*". IEEE Transactions on Circuits and Systems for Video Techonolgy, vol. 14, No. 1, 2004.

[Kanatani, 1984] KANATANI, K.. *"Detection of surface orientation and motion from texture by stereological technique*". Journal of Artificial Intelligence, vol. 23, pp. 213–237, 1984.

[Karu y Jain, 1996] KARU, K. JAIN, A. *"Fingerprint Classification*". Pattern Recognition, pp. 389–404, 1996.

[Kass y Witkin, 1987] KASS, M; WITKIN, A. *"Analyzing oriented patterns"*. Computer Vision, Graphics, and Image Processing, vol. 37, No. 3, pp. 362-385, 1987.

[Kawagoe y Tojo, 1984] KAWAGOE, M; TOJO, A. *"Fingerprint Pattern Classification*". Pattern Recognition, pp. 295–303, 1984.

[Kaymaz y Mitra, 1993] KAYMAZ, E.; MITRA. S. *"A Novel Approach to*

Fourier Spectral Enhacement of Laser-Luminescent Fingerprint Images". Journal of Forensic Sciences, vol. 38, No. 3, pp. 530, 1993.

[Keller y Hunt, 1985] KELLER J.; HUNT D. "*Incorporating fuzzy membership function into the perceptron algorithm*". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. PAMI-7, pp. 693-699.

[Khuwaja y Abu-Rezq, 2003] KHUWAJA, G.; ABU-REZQ, N. "*Data acquisition and recognition of fingerprints with LVQ*". International Journal of Computational Intelligence and Applications vol. 3, No. 1, pp 65-88, 2003.

[Kohonen, 1982] KOHONEN, T. *"Self-organized formation of topologically correct features maps*". Biological Cybernetics, vol. 43, pp. 59-69, 1982.

[Kohonen, 1982a] KOHONEN, T. *"Análisis of a simple self-organizing process*". Biological Cybernetics, vol. 44, pp. 135–140, 1982.

[Kung, 1993] KUNG, S. *"Digital Neural Networks*". Prentice Hall Inc. New Yersey, 1993.

[Leach, 2000] LEACH, S. "*Singular Value Decomposition-A Primer*". Department of Computer Science Brown University, Providence, EEUU, 2000.

[Lim, 1990] LIM, J. "*Two-Dimensional Signal and Image Processing*". Prentice Hall Inc. New Yersey, 1990.

[Lindeberg y Garding , 1993] LINDEBERG, T.; GARDING, J. *"Shape from texture from a multi-scale perspective"*. Proceedings 4th International Conference on Computer Vision, IEEE Computer Society Press, Berlin, Germany, pp. 683–691, 1993.

[Lobato et al., 2000a] LOBATO, E.; SEARA, R.; ROCHA, C. "*Identificação de impressões digitais via transformada wavelet*". CBA 2000-XIII Congreso Brasileiro de Automática, Florianópolis, Brasil, pp.1415-1420, 2000.

[Lobato y Seara, 2000b] LOBATO, E.; SEARA, R. "*Clasificação e Identificação de impressões digitais em tempo real de análisis multi-resolução da informação direcional*". SBrT 2000-XVIII Simpósio Brasileiro de Telecomunicaçães, Gramado, Brasil, pp. 612-617, 2000.

[Lorenzi, 2000] LORENZI, S. *"DSP Nuevas arquitecturas y mayor potencia de cálculo"*. Universidad Politécnica de Cataluña, Revista Mundo Electrónico, pp. 78-83, 2000.

[Maio y Maltoni, 1997] MAIO, D.; MALTONI, D. "*Direct Gray-Scale Minutiae Detection in Fingerprints"*. IEEE Transactions on Pattern Análisis and Machine Intelligence, vol 19, no. 1, 1997.

[Maio y Maltoni, 1998] MAIO, D.; MALTONI, D. "*Ridge-Line Density Estimation in Digital Images*". In Proc. Int. Conference on Pattern Recognition (14th), pp. 534 - 538, 1998.

[Maio et al., 2000] MAIO, D; MATLTONI, D.; CAPELLI, R. WAYMAN, J.L.; JAIN A. "*FVC2000: Fingerprint Verification Competition*". Technical Reports: DEIS, Universitiy of Bologna, Setiembre, 2000.

[Maio et al., 2002a] MAIO, D.; MALTONI, D.; CAPELLI, R. WAYMAN, J.L.; JAIN A.K. "*FVC2000: Fingerprint Verification Competition*". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, No 3, pp. 402 - 412, 2002.

[Maio et al., 2002b] MAIO, D.; MALTONI, D.; CAPELLI, R.; WAYMAN, J.; JAIN A. "*FVC2002: Second Fingerprint Verification Competition*", in Proc. Int. Conf. on Pattern Recognition (16th), vol. 3, pp. 811 -

814, 2002.

[Maltoni et al., 2003] MALTONI, D.; MAIO, D.; JAIN, A.; PRABHAKAR, S. "*Handbook of Fingerprint Recognition*". Springer-Verlag, First edition, 2003.

[Masters, 1994] MASTERS T. "*Signal and Image Processing With Neural Networks: A C++ Sourcebook*". Wiley, 1994.

[Mathews y Fink, 1999] MATHEWS, J.; FINK, C. *"Métodos Numericos con Matlab"*. Prentice Hall, Tercera edición, 1999.

[Mehtre et al ,1987] MEHTRE, B.; MURTHY, N.; KAPOOR, S.; CHATTERJEE, B. "*Segmentation of Fingerprint Images Using the Directional Image*". Pattern Recognition, vol. 20, No. 4, pp. 429 – 435, 1987.

[Mehtre y Chatterjee, 1989] MEHTRE, B.; CHATTERJEE, B. "*Segmentation of Fingerprint Images A Composite Method*". Pattern Recognition, vol. 22, No. 4, pp. 381 – 385, 1989.

[O'Neil, 1994] O'NEIL, P. *"Matemáticas Avanzadas para Ingenería"*. CECSA, Primera edición en Español, 1994.

[Oppenheim et al., 2000] OPPENHEIM, A.; SCHAFER, R.; BUCK, J. *"Tratamiento de señales en tiempo discreto"*. Prentice Hall, Segunda edición en Español, 2000.

[Pajares y De la Cruz, 2002] PAJARES G.; DE LA CRUZ, J. *"Visión por computador"*. Editorial RA-MA , Primera edición, 2002.

[Pandya y Macy, 1996] PANDYA, A.; MACY, R. *"Pattern Recognition with Neural Networks in C++"*. CRC and IEEE Press, 1996.

[Pérez, 2002] PÉREZ C. "*Matlab y sus Aplicaciones en las Ciencias*

y la Ingeniería". Prentice Hall, 2002.

[Perona, 1999] PERONA, P. "*Orientation diffusions*". IEEE Transactions on Image Processing , vol.20, No. 1, pp. 100-107, 1999.

[Polikarpova, 1996] POLIKARPOVA, N. "*On the Fractal Feature in Fingerprint Analysis*". In Proceedings Int. Conference on Pattern Recognition (13th), vol. 3, pp. 591–595, 1996.

[Plaz y Lang, 2004] PLAZ M.; LANG A. "*Fingerprint Authentication System*". HSR Hochschule fur Technik Rapperswil, 2004.

[Prabhakar, 2001]. PRABHAKAR, S. "*Fingerprint Classification and Matching Using a Filterbank*". PhD Thesis, *Michigan State University*, Michigan USA, 2001.

[Rademacher, 2001] RADEMACHER, D. *"Face Detection"*. Colorado School of Mines, 2001.

[Ramos, 1997] RAMOS G. "*La compresión de imágenes de impresiones dactilares mediante Wavelets*" Tesis de Licenciatura, Departamento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, 1997.

[Ratha et al., 1995] RATHA, N.; CHEN, S.; JAIN, A. "*Adaptative Flow Orientation – Based Feature Extraction in Fingerprint Images*". Pattern Recognition, vol. 28, nº 11, pp. 1657-1672, 1995.

[Ratha et al., 1996] RATHA, N.; KARU, K.; CHEN, S.; JAIN, A. "*A Real Time Matching System for Large Fingerprint Databases*". IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, nº 8, pp. 799-813, 1996.

[Salas, 2002] SALAS, R. *"Robustez en Redes Neuronales Feedforward"*. Tesis de Maestría en Ingenería Informática,

Universidad tecnica Federico Santa María, Valparaíso - Chile, 2002.

[Serrat et al., 2000] SERRAT, J.; LOPEZ, A. ; LLORET, D. "*On ridges and valleys*". Pattern Recognition International Conference.(15th), Barcelona , España, vol. 4, pp. 59-66, 2000.

[Shen y Khanna, 1994] SHEN, W.; KHANNA, R. "*Automated Fingerprint Indentification Systems (AFIS) Bencmarking Using the National Institute of Standars and Technology (NIST) Special Database 4*". In Proceedings Int. Carnahan Conference on Security Technology (28th), pp 88 - 95, 1994.

[Stosz y Alyea., 1994] STOSZ, J.; ALYEA, L. *"Automated System for Fingerprint Authentication Using Pores and Ridge Structure "*. In Proc. of SPIE (Automatic Systems for the Identification and Inspection of Humans), vol. 2277, pp. 210-223, 1994.

[Svarer, 1995] SVARER, C. *"Neural Networks for Signal Processing"*. PhD. Thesis Electronics Institute Technical University of Denmark, Mayo, 1995.

[Takeda et al., 1990] TAKEDA, M.; UCHIDA, S. ; HIRAMATSU, K.; MATSUNAMI, T. *"Identification Method for Personal Verification"*. In Proceedings Int. Conference Pattern Recognition (10th), vol. 1, pp. 761–766, 1990.

[Texas, 2001a] TEXAS INSTRUMENTS "*TMS320C6000 CPU and Instruction Set Reference Guide*". (SPRU189F) USA, 2001.

[Texas, 2001b] TEXAS INSTRUMENTS "*TMS320C6000 Peripherals Reference Guide*". (SPRU190D) USA, 2001.

[Texas, 2001c] TEXAS INSTRUMENTS "*Code Composer Studio - Users Guide*". (SPRU509C) USA, 2001.

[Texas, 2001d] TEXAS INSTRUMENTS "*TMS320C6000 DSP/BIOS User's Guide*". (SPRU423A) USA, 2001.

[Texas, 2001e] TEXAS INSTRUMENTS "*TMS320C6000 Programmer's Guide*". (SPRU198F) USA, 2001.

[Texas, 2001f] TEXAS INSTRUMENTS "*TMS320C6000 Optimizing Compiler User's Guide*". (SPRU187I) USA, 2001

[Tico et al., 2001a] TICO, M.; ONNIA, V.; KUOSMANEN, P. *"Fingerprint Image Enhancement based on Second Directional Derivate of the Digital Image"*. Submitted to EURASIP Journal on Applied Signal Processing, 2001.

[Tico et al., 2001b] TICO, M.; ONNIA, V.; SAARINEN, J.; RAMÖ, P. *"Optimized singular point detection algorithm for fingerprint Images"*. In Proceedings of ICIP (International Conference on Image Processing), Greece, 2001.

[Tico y Onnia, 2002], TICO, M.; ONNIA, V. *"Adaptive Binarization method for Fingerprint Images"*. IEEE International Conference on Acoustics, Speech and Signal Processing, vol. 4, pp. 3692-3695, Florida - USA, 2002.

[Umeyama, 1991] UMEYAMA, S. *"Least-Square Estimation of Transformation Parameters Between Two Point Patterns"*. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, No. 4, pp. 376-380, 1991.

[Verma et al., 1987] VERMA, M.; MAJUNDAR, A.; CHATTERJEE. B. "*Edge detection in Fingerprint*". Pattern Recognition, vol. 20, nº 5, pp. 513-523, 1987.

[Watson et al., 1994] WATSON, C. I. ; CANDELA, G.I. ; GROTHER P.J. "*Comparison of FFT Fingerprint Filtering Methods for Neural*

Network Classification". Technical Report: NIST TR 5493, Sept, 1994.

[Willis y Myers, 2001] WILLIS, A. J. ; MYERS, L. "*A cost-effective fingerprint recognition system for use with low-quality prints and damage fingertips*". Pattern Recognition, vol. 34, No 2, pp. 233–270, 2001.

[Xiao y Bian, 1986] XIAO, X.; BIAN, Z. *"An Approach to Fingerprint Identification by Using the Attributes of Features Lines of Fingerprint"*. In Proceedings Int. Conference Pattern Recognition (8th), pp. 663–665, 1986.

[Xiping et al., 2000] XIPING, L.; JIE,T.; YAN, W. *"A Minutia Matching Algorithm in Fingerprint Verification"*. International Conference on Pattern Recognition (ICPR'00), vol. 4, pp. 833–836, Barcelona, España, 2000.

[Ying et al, 2002] YING, H.; TIENIU, T.; YUNHONG, W. "*Fingerprint matching based on error propagation*". Proceedings International Conference on Image Processing, vol. 1, pp. 273–276, 2002.

[Zang y Suen, 1984] ZANG, T. ; SUEN, C. *"A fast parallel algorithm for thinning digital patterns*". Communications of the ACM, vol. 27, pp. 236-239, 1984.

[Zhang et al., 2001] ZHANG, Q.; HUANG, K.; YAN, H. *"Fingerprint Classification Based on Extraction and Analysis of Singularities and Pseudoridges*". Pan Sydney Area Workshop on Visual Information Processing (VIP2001), Sydney, Australia, 2001.

[Zhao y Tang, 2002] ZHAO, F. ; TANG, X. *"Preprocessing for skeleton-based fingerprint minutiae extraction*". CISST 02 International Conference, 2002.

Referencias URL

[URL 1.1] http://www.biometricgroup.com/reports/public/market_report.html. Acceso el 20 de Julio del 2004

[URL 5.1] http://astronomy.swin.edu.au/~pbourke/analysis/correlate/. Acceso el 20 de Julio del 2004

[URL 5.2] http://www.owlnet.rice.edu/~elec431/projects96/pictomaniacs/DSP.html. Acceso el 20 de Julio del 2004.

[URL 5.3] LEWIS; J. "Fast Normalized Cross Correlation", http://www.idiom.com/~zilla/Work/nvisionInterface/nip.html, Acceso el 24 Mayo del 2004

[URL 5.4] http://www.qhull.org , Acceso el 26 Mayo del 2004

[URL 5.5] http://www.biometrika.it/eng/, Acceso el 3 de Agosto del 2004.

[URL 6.1] http://www.codeproject.com/bitmap/cximage.asp. Acceso el 15 de Enero del 2002.

[URL 6.2] http://www.cvmt.dk/~hn/Images/install/IPL/. Acceso el 20 de Enero del 2002.

Apéndice A
Tabla para la búsqueda de rutas

	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	2	1	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	3	0	1	0	0	0	0	0
6	0	3	2	0	0	0	0	0
7	0	0	1	0	0	0	0	0
8	0	0	0	0	0	0	0	0
9	4	0	0	1	0	0	0	0
10	0	4	0	2	0	0	0	0
11	4	0	0	2	0	0	0	0
12	0	0	4	3	0	0	0	0
13	4	0	0	1	0	0	0	0
14	0	4	0	2	0	0	0	0
15	4	0	0	2	0	0	0	0
16	0	0	0	0	0	0	0	0
17	5	0	0	0	1	0	0	0
18	0	5	0	0	2	0	0	0
19	5	5	0	0	2	0	0	0
20	0	0	5	0	3	0	0	0
21	5	0	0	0	1	0	0	0
22	0	5	0	0	2	0	0	0
23	5	5	0	0	2	0	0	0
24	0	0	0	5	4	0	0	0
25	4	0	0	1	1	0	0	0
26	0	0	0	2	2	0	0	0
27	4	0	0	2	2	0	0	0
28	0	0	0	0	3	0	0	0
29	4	0	0	1	1	0	0	0
30	0	0	0	2	2	0	0	0
31	4	0	0	2	2	0	0	0
32	0	0	0	0	0	0	0	0
33	6	0	0	0	0	1	0	0
34	0	6	0	0	0	2	0	0
35	6	6	0	0	0	2	0	0

	1	2	3	4	5	6	7	8
36	0	0	6	0	0	3	0	0
37	6	0	6	0	0	3	0	0
38	0	6	6	0	0	2	0	0
39	6	6	6	0	0	2	0	0
40	0	0	0	6	0	4	0	0
41	0	0	0	1	0	0	0	0
42	0	6	0	0	0	2	0	0
43	0	6	0	0	0	2	0	0
44	0	0	6	0	0	4	0	0
45	0	0	6	1	0	0	0	0
46	0	6	6	0	0	2	0	0
47	0	6	6	0	0	2	0	0
48	0	0	0	0	6	5	0	0
49	6	0	0	0	1	1	0	0
50	0	6	0	0	2	2	0	0
51	6	6	0	0	2	2	0	0
52	0	0	6	0	0	3	0	0
53	6	0	6	0	1	3	0	0
54	0	6	6	0	2	2	0	0
55	6	6	6	0	2	2	0	0
56	0	0	0	6	0	4	0	0
57	0	0	0	1	1	0	0	0
58	0	6	0	0	2	2	0	0
59	0	6	0	0	2	2	0	0
60	0	0	6	0	0	4	0	0
61	0	0	6	1	1	0	0	0
62	0	6	6	0	2	2	0	0
63	0	6	6	0	2	2	0	0
64	0	0	0	0	0	0	0	0
65	7	0	0	0	0	0	1	0
66	0	7	0	0	0	0	2	0
67	0	7	0	0	0	0	2	0
68	0	0	7	0	0	0	3	0
69	0	0	7	0	0	0	3	0
70	0	7	7	0	0	0	2	0
71	0	7	7	0	0	0	2	0
72	0	0	0	7	0	0	4	0
73	4	0	0	1	0	0	4	0
74	0	7	0	0	0	0	0	0
75	4	7	0	0	0	0	0	0
76	0	0	7	7	0	0	4	0
77	4	0	7	1	0	0	4	0
78	0	7	7	0	0	0	0	0
79	4	7	7	0	0	0	0	0

	1	2	3	4	5	6	7	8
80	0	0	0	0	7	0	5	0
81	5	0	0	0	1	0	0	0
82	0	7	0	0	2	0	2	0
83	5	7	0	0	2	0	2	0
84	0	0	7	0	0	0	3	0
85	5	0	7	0	1	0	3	0
86	0	7	7	0	2	0	2	0
87	5	7	7	0	2	0	2	0
88	0	0	0	7	0	0	4	0
89	4	0	0	1	1	0	4	0
90	0	7	0	0	2	0	0	0
91	4	7	0	0	2	0	0	0
92	0	0	7	7	0	0	4	0
93	4	0	7	1	1	0	4	0
94	0	7	7	0	2	0	0	0
95	4	7	7	0	2	0	0	0
96	0	0	0	0	0	7	6	0
97	6	0	0	0	0	1	0	0
98	0	6	0	0	0	2	2	0
99	6	6	0	0	0	2	2	0
100	0	0	6	0	0	3	3	0
101	6	0	6	0	0	3	3	0
102	0	6	6	0	0	2	2	0
103	6	6	6	0	0	2	2	0
104	0	0	0	0	0	4	4	0
105	0	0	0	1	0	0	4	0
106	0	6	0	0	0	2	0	0
107	0	6	0	0	0	2	0	0
108	0	0	6	0	0	4	4	0
109	0	0	6	1	0	0	4	0
110	0	6	6	0	0	2	0	0
111	0	6	6	0	0	2	0	0
112	0	0	0	0	0	0	5	0
113	6	0	0	0	1	1	0	0
114	0	6	0	0	2	2	2	0
115	6	6	0	0	2	2	2	0
116	0	0	6	0	0	3	3	0
117	6	0	6	0	1	3	3	0
118	0	6	6	0	2	2	2	0
119	6	6	6	0	2	2	2	0
120	0	0	0	0	0	4	4	0
121	0	0	0	1	1	0	4	0
122	0	6	0	0	2	2	0	0
123	0	6	0	0	2	2	0	0

	1	2	3	4	5	6	7	8
124	0	0	6	0	0	4	4	0
125	0	0	6	1	1	0	4	0
126	0	6	6	0	2	2	0	0
127	0	6	6	0	2	2	0	0
128	0	0	0	0	0	0	0	0
129	8	0	0	0	0	0	0	1
130	0	8	0	0	0	0	0	2
131	0	8	0	0	0	0	0	2
132	0	0	8	0	0	0	0	3
133	0	0	8	0	0	0	0	3
134	0	8	8	0	0	0	0	0
135	0	8	8	0	0	0	0	0
136	0	0	0	8	0	0	0	4
137	4	0	0	8	0	0	0	4
138	0	0	0	8	0	0	0	4
139	4	0	0	8	0	0	0	4
140	0	0	8	8	0	0	0	4
141	4	0	8	8	0	0	0	4
142	0	0	8	8	0	0	0	4
143	4	0	8	8	0	0	0	4
144	0	0	0	0	8	0	0	5
145	5	0	0	0	8	0	0	5
146	0	0	0	0	0	0	0	5
147	5	0	0	0	0	0	0	5
148	0	0	8	0	8	0	0	5
149	5	0	8	0	8	0	0	5
150	0	0	8	0	0	0	0	5
151	5	0	8	0	0	0	0	5
152	0	0	0	8	8	0	0	4
153	4	0	0	8	8	0	0	4
154	0	0	0	8	0	0	0	4
155	4	0	0	8	0	0	0	4
156	0	0	8	8	8	0	0	4
157	4	0	8	8	8	0	0	4
158	0	0	8	8	0	0	0	4
159	4	0	8	8	0	0	0	4
160	0	0	0	0	0	8	0	6
161	6	0	0	0	0	0	0	6
162	0	6	0	0	0	2	0	0
163	6	6	0	0	0	2	0	0
164	0	0	0	0	0	3	0	0
165	6	0	0	0	0	3	0	0
166	0	6	0	0	0	2	0	0
167	6	6	0	0	0	2	0	0

	1	2	3	4	5	6	7	8
168	0	0	0	8	0	0	0	4
169	0	0	0	8	0	0	0	4
170	0	6	0	8	0	2	0	4
171	0	6	0	8	0	2	0	4
172	0	0	0	8	0	0	0	4
173	0	0	0	8	0	0	0	4
174	0	6	0	8	0	2	0	4
175	0	6	0	8	0	2	0	4
176	0	0	0	0	8	0	0	6
177	6	0	0	0	8	0	0	6
178	0	6	0	0	0	2	0	0
179	6	6	0	0	0	2	0	0
180	0	0	0	0	8	3	0	0
181	6	0	0	0	8	3	0	0
182	0	6	0	0	0	2	0	0
183	6	6	0	0	0	2	0	0
184	0	0	0	8	8	0	0	4
185	0	0	0	8	8	0	0	4
186	0	6	0	8	0	2	0	4
187	0	6	0	8	0	2	0	4
188	0	0	0	8	8	0	0	4
189	0	0	0	8	8	0	0	4
190	0	6	0	8	0	2	0	4
191	0	6	0	8	0	2	0	4
192	0	0	0	0	0	0	8	7
193	7	0	0	0	0	0	0	0
194	0	8	0	0	0	0	2	0
195	0	8	0	0	0	0	2	0
196	0	0	8	0	0	0	3	3
197	0	0	8	0	0	0	3	3
198	0	8	8	0	0	0	2	0
199	0	8	8	0	0	0	2	0
200	0	0	0	8	0	0	4	4
201	4	0	0	8	0	0	4	4
202	0	0	0	8	0	0	0	4
203	4	0	0	8	0	0	0	4
204	0	0	8	8	0	0	4	4
205	4	0	8	8	0	0	4	4
206	0	0	8	8	0	0	0	4
207	4	0	8	8	0	0	0	4
208	0	0	0	0	8	0	0	5
209	5	0	0	0	8	0	0	5
210	0	0	0	0	0	0	2	5
211	5	0	0	0	0	0	2	5

	1	2	3	4	5	6	7	8
212	0	0	8	0	8	0	3	5
213	5	0	8	0	8	0	3	5
214	0	0	8	0	0	0	2	5
215	5	0	8	0	0	0	2	5
216	0	0	0	8	8	0	4	4
217	4	0	0	8	8	0	4	4
218	0	0	0	8	0	0	0	4
219	4	0	0	8	0	0	0	4
220	0	0	8	8	8	0	4	4
221	4	0	8	8	8	0	4	4
222	0	0	8	8	0	0	0	4
223	4	0	8	8	0	0	0	4
224	0	0	0	0	0	8	0	6
225	6	0	0	0	0	0	0	6
226	0	6	0	0	0	2	2	0
227	6	6	0	0	0	2	2	0
228	0	0	0	0	0	3	3	0
229	6	0	0	0	0	3	3	0
230	0	6	0	0	0	2	2	0
231	6	6	0	0	0	2	2	0
232	0	0	0	8	0	0	4	4
233	0	0	0	8	0	0	4	4
234	0	6	0	8	0	2	0	4
235	0	6	0	8	0	2	0	4
236	0	0	0	8	0	0	4	4
237	0	0	0	8	0	0	4	4
238	0	6	0	8	0	2	0	4
239	0	6	0	8	0	2	0	4
240	0	0	0	0	8	0	0	6
241	6	0	0	0	8	0	0	6
242	0	6	0	0	0	2	2	0
243	6	6	0	0	0	2	2	0
244	0	0	0	0	8	3	3	0
245	6	0	0	0	8	3	3	0
246	0	6	0	0	0	2	2	0
247	6	6	0	0	0	2	2	0
248	0	0	0	8	8	0	4	4
249	0	0	0	8	8	0	4	4
250	0	6	0	8	0	2	0	4
251	0	6	0	8	0	2	0	4
252	0	0	0	8	8	0	4	4
253	0	0	0	8	8	0	4	4
254	0	6	0	8	0	2	0	4
255	0	6	0	8	0	2	0	4