

# Verificación Off-Line de Firmas Manuscritas: Una Propuesta basada en Snakes Paramétricos

José F. Vélez

Ángel Sánchez

Ana B. Moreno

José L. Esteban

Grupo de Algorítmica para la Visión Artificial y la Biometría (GAVAB), Escuela Superior de Ciencias Experimentales y Tecnología, Departamento de Informática, Estadística y Telemática, Universidad Rey Juan Carlos, 28933, Madrid.

[jose.velez@urjc.es](mailto:jose.velez@urjc.es), [angel.sanchez@urjc.es](mailto:angel.sanchez@urjc.es), [belen.moreno@urjc.es](mailto:belen.moreno@urjc.es), [joseluis.esteban@urjc.es](mailto:joseluis.esteban@urjc.es)

## Resumen

En este artículo se presenta un método mejorado basado en la propuesta de *snakes* realizada por Kass, Witkin y Terzopoulos [11]. Dicho algoritmo permite la verificación, frente a falsificadores no entrenados, utilizando una firma en forma de imagen 2D bitonal obtenida *off-line*. Para la construcción del sistema se utiliza una única firma por individuo como muestra de aprendizaje. Además, se describe el caso real del problema de verificación de firmas en entorno bancario, y se adaptan los parámetros del algoritmo para resolver este problema de manera efectiva y eficiente.

## 1. Introducción

Las firmas manuscritas constituyen uno de los principales métodos de justificación de autoría que manejamos las personas [12]. Una de sus ventajas radica en la facilidad de realización, pues apenas se necesita un papel y un bolígrafo. Otra ventaja está en la dificultad de realizar una falsificación que pueda engañar a una persona experta. Pero quizás su mayor ventaja esté en que es un mecanismo de autenticación personal ampliamente aceptado por la sociedad, en general, y por organismos públicos y privados, en particular.

Es por estas razones que la construcción de un sistema que realice la tarea de verificación de firmas de manera automática suscita un enorme interés [10]. El problema se ha abordado desde múltiples enfoques, habiéndose logrado resultados esperanzadores bajo condiciones controladas. Sin embargo, aún estamos lejos de un sistema que, en condiciones reales, realice esta tarea de manera automática con la efectividad que la realiza una persona con un entrenamiento mínimo.

El apartado 2 describe el problema de verificación. El apartado 3 realiza una breve introducción a los *snakes* 2D sobre imágenes estáticas. El apartado 4 introduce el algoritmo

propuesto basado en una modificación al algoritmo original de *snakes*. El apartado 5 muestra nuestros experimentos. Finalmente, el apartado 6 presenta las conclusiones y propone futuras extensiones al trabajo realizado.

## 2. La verificación de firmas

Supongamos que a un sistema que conoce las firmas de un conjunto de sujetos, se le presenta una firma y la supuesta identidad del firmante. El problema de la verificación consiste en que el sistema determina el grado de similitud entre la firma presentada y las que conoce del auténtico firmante, para establecer si es auténtica o no. Como enuncia E. Justino [10] “en el problema de verificación de firmas se trata de maximizar las diferencias interpersonales y minimizar las diferencias intrapersonales”.

Los falsificadores, que pueden tratar de engañar a un sistema de este tipo, pueden catalogarse en dos grupos: los falsificadores entrenados y los no entrenados [12]. Los primeros, que conocen la firma de la persona que quieren suplantar, se entrenan en reproducir la firma y consiguen unas falsificaciones de gran calidad. Los segundos no sólo no están entrenados, sino que no han visto nunca la firma, por lo que su reproducción no guarda ningún parecido con la auténtica. Curiosamente, y en contra de lo que pudiese pensarse, a este segundo grupo de falsificadores corresponde el 95% del fraude existente en entidades bancarias en el mundo [8].

El proceso de captura de firmas puede realizarse de dos formas diferentes: *on-line* y *off-line* [19]. En el modo *on-line* la captura de la firma se realiza utilizando un dispositivo especial (clásicamente una lápiz electrónico y una tableta gráfica especial) que recoge información dinámica del escritor durante el acto de firmar. Esta información incluye, además del grafismo, datos como la presión, la velocidad, los puntos de inicio,

las direcciones de los trazos, la inclinación, etc [19]. Existen actualmente sistemas que realizan procesos muy precisos de reconocimiento y verificación utilizando esta información. El método de captura *off-line* se basa en el escaneo de una firma, una vez realizada sobre un soporte ordinario (clásicamente papel). En este caso, la información de que se dispone es mucho menor y la resolución espacial y radiométrica a la que se escanee la firma son elementos que pueden influir en la verificación.

Además, en la formulación *off-line*, aparece el problema de localizar y segmentar la firma dentro del documento. A su vez, la segmentación de la firma dentro de un documento general presenta diferentes problemas: desconocimiento de la posición exacta de la firma, existencia de ruido blanco y de ruido con estructura. La Figura 1 ilustra algunos de estos problemas: sellos que se han superpuesto a la firma (c) y (d), tramas de logotipos sobre los que se ha firmado (b), texto adyacente a la zona de firmado (a) y líneas o cuadros sobre los que se firma (a).

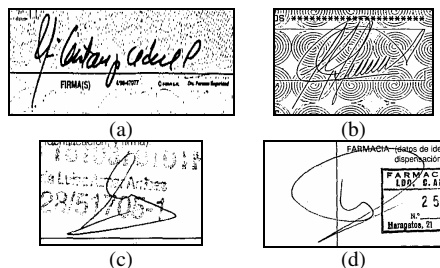


Figura 1. Ejemplos de ruido blanco (a) y estructural (b), (c) y (d) que dificultan la segmentación de la firma.

## 2.1. Requisitos de un sistema real

Un primer requisito de un sistema de verificación, utilizable en condiciones reales, consiste en que no es viable solicitar muchas firmas a cada usuario para construir el sistema. En general sólo suele disponerse de una firma por persona. Sin embargo, sí es posible que la firma que se utilice para construir el sistema esté libre de ruido y en una posición conocida.

Otro requisito consiste en que, un sistema real de verificación de firmas, deberá resultar escalable con respecto al número de individuos que es capaz de verificar, y no deberá ser exigente en cuanto a

las condiciones de resolución tanto espacial como radiométrica.

Además, el usuario de un sistema como el indicado desea percibir que el sistema sea fiable ante errores y resulte rápido en su funcionamiento. En algunos ámbitos, como el bancario, es preferible que el sistema acepte una firma falsa (Tasa de Error al Aceptar o FAR) a que rechace una firma verdadera (Tasa de Error al Rechazar o FRR) siempre que el coste económico de la supuesta falsificación no supere un umbral. Esto se debe a que previamente a la introducción de un sistema de verificación no se analiza la firma, o sólo se hace en casos de importes elevados, y por lo tanto el coste de trabajo es cero. Elevar este coste no suele ser aceptado por un cliente que “antes no tenía este problema”.

## 2.2. Trabajos existentes

El tratamiento de firmas es un área de investigación muy activa desde mediados de 1970 [12]. Existen multitud de trabajos que abordan cada uno de los aspectos que se han presentado. Por ejemplo los hay que tratan el problema de la localización de la firma en un entorno ruidoso [18], los que tratan el problema de la imposibilidad de usar más de una firma por individuo para el entrenamiento [7], los que tratan el problema de la escalabilidad no asistida [4], o que tratan el problema de falsificadores entrenados [6].

En general, los enfoques existentes comparten el esquema clásico de extracción de características discriminantes y el posterior uso de un clasificador basado en tales características. Entre estas últimas las más utilizadas son: centros de gravedad (parciales o totales, línea base global (*global base line*), los límites superiores e inferiores de la firma, número de agujeros de la firma, esqueleto de la firma, tamaño de la envolvente convexa, caja que contiene la firma (*bounding box*), contorno o perímetro, ejes de mayor y menor inercia, relación entre área y perímetro, densidad de puntos en las regiones de la imagen, ángulo de inclinación (*slant*), puntos extremos superiores e inferiores, número de trazos, estructura de los trazos, puntos de cruce y de relleno de la firma a partir del eje de mínima inercia.

En cuanto a los métodos de clasificación existen propuestas variadas: los que usan HMMs [10][4], funciones de desplazamiento óptimo [7], ajuste elástico [6], lógica borrosa [9], redes neuronales [2][15] y algoritmos genéticos [14].

### 3. Los snakes

Los *snakes* [11][5] son un tipo de modelos de contornos activos [7], que se basan en el estudio del movimiento de un contorno abierto o cerrado sobre una imagen a la que trata de adaptarse. Asociado a este contorno se define una función de energía que tiene una componente interna y otra externa. La componente interna de la energía se debe a ciertas características de elasticidad y flexibilidad con que se dota al contorno. La componente externa se debe a la influencia de la imagen sobre la que se mueve el contorno. Utilizando estas energías, un algoritmo mueve el contorno sobre la imagen buscando una posición de mínima energía. En este movimiento de búsqueda del mínimo se aprecia como serpentea el contorno, hecho que da nombre al algoritmo.

Representando el parámetro  $s$  la posición del *snake* sobre una imagen bidimensional como un conjunto infinito de puntos  $v(s)=(x(s),y(s))$  se puede escribir la función de energía como:

$$E_{snake} = \int_0^1 E(v(s)) ds = \int_0^1 [E_{int}(v(s)) + E_{imagen}(v(s)) + E_{cons}(v(s))] ds \quad (1)$$

En esta fórmula  $E_{int}$  representa una energía debida a la relación entre los puntos del *snakes*,  $E_{imagen}$  representa una energía relativa a la posición que ocupan los puntos del *snakes* dentro de la imagen, y finalmente  $E_{cons}$  representa una energía asociada a otras condiciones externas.

#### 3.1. Energía interna

Clásicamente el término  $E_{cons}$  no se considera, y para  $E_{int}$  se propone la siguiente ecuación [11]:

$$E_{int} = \frac{1}{2} (\alpha(s)[x_s^2(s) + y_s^2(s)] + \beta(s)[x_{ss}^2(s) + y_{ss}^2(s)]) \quad (2)$$

En la fórmula,  $x_s$  y  $y_s$  representan la derivada primera respecto al parámetro de posición  $s$ . Igualmente,  $x_{ss}$  y  $y_{ss}$  representan la derivada segunda. Por otro lado, los coeficientes  $\alpha(s)$  y  $\beta(s)$

corresponden respectivamente a la importancia de la elasticidad y a la flexibilidad del *snakes*. Valores altos de  $\alpha(s)$  hacen al *snakes* encoger su forma. Mientras, valores altos de  $\beta(s)$  fomentan que el *snakes* enderecen su forma si no es cerrado, o se vuelva convexo si es cerrado, haciéndolo en general más suave y menos anguloso.

Otras formulaciones [5] han sido propuestas para solventar diferentes tipos de problemas relacionados con el comportamiento del *snake* ante problemas particulares. En general, casi todas las formulaciones se basan en ecuaciones de tipo muelle.

#### 3.2. Energía de la imagen

Asociado a cada punto de la imagen sobre la que se mueve el *snake* se define una función de energía. La función de energía en cada punto permite puntuar con valores menores aquellos puntos donde se desea que se sitúe el *snake*. Además se suele dotar a los puntos de una imagen de valores en gradiente que convergen hacia puntos de mínima energía, para fomentar el movimiento del *snake* en dicha dirección.

En su formulación original, el algoritmo de *snakes* propuesto por Kass y otros [11], contempla términos que atraen al contorno activo hacia líneas, bordes y terminaciones. Estos accidentes topológicos deben ser previamente segmentados en la imagen sobre la que se sitúa el *snake*. Así, la energía de la imagen se formula como:

$$E_{imagen} = E_{líneas} + E_{bordes} + E_{term}$$

Así, para atraer el *snake* hacia las líneas Kass propone utilizar la función de la imagen por sí misma.

$$E_{líneas} = w I(x,y) \quad (3)$$

Para atraer el *snake* hacia los bordes se propone la función gradiente.

$$E_{bordes} = -|\nabla I(x,y)|^2$$

Finalmente, para atraer el *snake* hacia los puntos de terminación se propone utilizar la curvatura de las líneas de nivel sobre la imagen suavizada  $C$  mediante un filtro de Gauss.

$$E_{term} = \frac{\partial \theta}{\partial n_{\perp}}$$

donde  $\theta = \arctg(Cy/Cx)$  y  $n_{\perp} = (-\sin(\theta), \cos(\theta))$

### 3.3. Minimización de la energía del *snake*

En la búsqueda de un mínimo para el *snake* se hacen múltiples simplificaciones. En primer lugar se discretiza el contorno, de manera que pasa de ser una función continua a estar compuesto de un conjunto  $n$  de puntos de control  $\{v_i\}_{i=1}^n$  que forman parte de un *spline* o incluso de un simple polígono. En segundo lugar se hacen constantes los coeficientes para las energías internas ( $\alpha(s)$  y  $\beta(s)$ ). Finalmente, se transforma la ecuación (1) a una formulación iterativa que permite el cálculo de las sucesivas posiciones de los puntos de control del *snake* en el tiempo  $\{v_i(t)\}$ , obteniendo finalmente un *problema variacional*.

Amini [1] propone el uso de programación dinámica para evitar la costosa evaluación de las múltiples posibilidades asociadas a cada movimiento del *snake*, siendo éste el enfoque utilizado por nuestra implementación. En el método propuesto por Amini el cálculo de la energía de (1) puede descomponerse en etapas sucesivas según la siguiente formulación:

$$E(v_1, v_2, \dots, v_n) = E_1(v_1, v_2) + E_2(v_2, v_3) + \dots + E_{n-1}(v_{n-1}, v_n)$$

A su vez, el cálculo de cada término de energía puede descomponerse utilizando programación dinámica discreta en una secuencia de funciones de una variable  $s_i$  donde el conjunto  $\{s_i\}_{i=1}^{n-1}$  se obtiene según el siguiente sistema de ecuaciones de recurrencia:

$$\begin{aligned} s_1(v_2) &= \min_{v_1} E_1(v_1, v_2) \\ s_2(v_3) &= \min_{v_2} \{s_1(v_2) + E_2(v_2, v_3)\} \\ s_3(v_4) &= \min_{v_3} \{s_2(v_3) + E_3(v_3, v_4)\} \\ &\dots \end{aligned}$$

$$\min_{v_1, \dots, v_n} E(v_1, \dots, v_n) = \min_{v_{n-1}} \{s_{n-2}(v_{n-1}) + E_{n-1}(v_{n-1}, v_n)\}$$

Para poder evaluar no sólo el primer sumando sino toda la ecuación (2), hay que tener en cuenta que la derivada segunda involucra a un tercer vértice, por lo que la descomposición de energías debe hacerse según:

$$\begin{aligned} E(v_1, v_2, \dots, v_n) &= E_1(v_1, v_2, v_3) + E_2(v_2, v_3, v_4) + \\ &\dots + E_{n-1}(v_{n-2}, v_{n-1}, v_n) \end{aligned}$$

donde 
$$E_{i-1}(v_{i-1}, v_i, v_{i+1}) = E_{img}(v_i) + E_{int}(v_{i-1}, v_i, v_{i+1}) \quad (4)$$

Por ello, la secuencia de funciones  $\{s_i\}_{i=1}^{n-1}$  será de dos variables, y para el caso de la función (2) tendrá la siguiente forma:

$$s_i(v_{i+1}, v_i) = \min \left\{ s_{i-1}(v_i, v_{i-1}) + \alpha |v_i - v_{i-1}|^2 + \beta |v_{i+1} - 2v_i + v_{i-1}|^2 + E_{ext}(v_i) \right\}$$

### 4. Método propuesto de verificación de firmas basado en *snakes*

El problema de verificar una firma consiste en minimizar la energía de ajuste del *snake* asociado a la firma a comprobar con respecto a la firma original.

El algoritmo de verificación propuesto en este trabajo puede resumirse en las siguientes etapas:

- 1.- Crear de una línea poligonal  $P$ , con vértices equiespaciados, sobre el trazo de la imagen de una firma. Tal firma se utiliza como muestra de aprendizaje.
- 2.- Disponer  $P$  sobre una firma a verificar de manera aproximada.
- 3.- Luego, utilizar el algoritmo de *snakes* para ajustar  $P$  exactamente a la nueva imagen.
- 4.- Utilizando una medida del incremento de energía o de *ajuste elástico* respecto a la deformación que se produce en el *snake* tras la adaptación, obtener una medida de similitud entre la firma que dio origen al *snake* y la firma sobre la que se ha iterado. Este valor permitirá considerar la firma como autentica o falsa.

#### 4.1. Propuesta inicial de ajuste

En una primera aproximación se utilizó la definición original de Kass [11] tanto para las energías internas, como para la energía correspondiente a la imagen. Como las imágenes de firmas eran bitonales, la localización de bordes y de extremos no tenía sentido para ajustar el *snake*. Tan sólo el uso de la energía asociada a cada píxel (3) era necesario para atraer el *snake* hacia la imagen de la firma.

Para imágenes sencillas el resultado resultaba prometedor, pues el *snake*, en unas pocas iteraciones se ajustaba con bastante precisión a la curva propuesta. Sin embargo el modelo tenía dos inconvenientes: por un lado la localidad y por otro la pérdida de forma.

Si el *snake* no se encontraba muy cerca de la zona de cambio de valor de los píxeles de la imagen, el algoritmo fracasaba. Además, si no encontraba pronto la curva para ajustarse a ella, el *snake* perdía rápidamente su forma original. Para subsanar ambos problemas se decidió ampliar la zona de influencia de los bordes de la imagen mediante un suavizado de Gauss. En la Figura 2 se puede observar el resultado de este primer enfoque sobre un trazo sencilla.

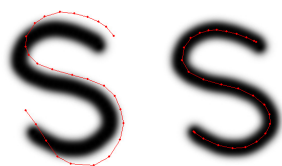


Figura 2. Ajuste del *snake* de parámetros  $(\alpha, \beta, \omega) = (0.1, 1.0, -10.0)$ . Donde  $\alpha, \beta, \omega$  aparecen en las ecuaciones (2) y (3).

El proceso seguido consistió en la creación manual de una línea poligonal  $P$  sobre el trazo de la imagen de una firma, considerada como muestra de aprendizaje. Además, se calcula el centro de masas  $C$  de los píxeles negros de la imagen de entrenamiento, y se localiza dentro del *snake*. Luego, se sitúa  $P$  sobre la imagen de una firma a verificar, haciendo coincidir el centro de masas de los píxeles negros de la imagen a verificar con el punto  $C$  del *snake*. Finalmente, se reescala  $P$  proporcionalmente a lo alto y a lo ancho para que coincida el ancho del *snake* con el ancho de la imagen.

Tras diversas pruebas se pudo comprobar que este primer enfoque sólo ayuda en la zona afectada por filtro de Gauss, lo cual resultó insuficiente para imágenes de firmas reales.

#### 4.2. Propuesta mejorada de ajuste

Para resolver estas dificultades se ha modificado la definición de energía  $E'_{snake}$  del algoritmo original de *snakes* en dos sentidos. Para solventar el problema de la localidad se utilizan *mapas de potencial* para la energía asociada a la imagen  $E'_{imagen}$ . Para evitar el problema de la pérdida de forma se proponen unas fuerzas internas  $E'_{forma}$  que mantienen la forma del *snake*. Así, la energía asociada al *snake* de una firma puede representarse como:

$$E'_{snake} = E'_{imagen} + E'_{forma}$$

#### Mapas de potencial

Cohen y Cohen [16] propusieron el uso de una fuerza externa utilizando un *mapa de potencial* basado en la distancia euclídea como solución al problema de no localidad. El enfoque parece óptimo para este caso, en el que los puntos pertenecientes a la firma están claramente determinados. El valor de distancia para cada punto del mapa  $m_{imagen}(x,y)$ , es igual al mínimo de los valores ya calculados para sus vecinos incrementado en una unidad. De esta forma, el algoritmo calcula los valores de distancia partiendo de los puntos adyacentes a la firma y avanzando en capas hacia los límites de la imagen. En la Figura 3 se aprecia el resultado del cálculo del *mapa de potencial* para una firma de ejemplo. El nivel de intensidad asociado a cada punto de la imagen representa la distancia al punto de la firma más cercano por lo que la diferencia entre dos píxeles adyacentes siempre es la unidad. Utilizando este *mapa de potencial* la energía de la imagen asociada a un punto de control  $v_i$  del *snake* se define como:

$$E'_{imagen} = m_{imagen}(v_{i-1}) + m_{imagen}(v_i) + m_{imagen}(v_{i+1})$$

donde  $i$  varía entre 1 y  $n$ .

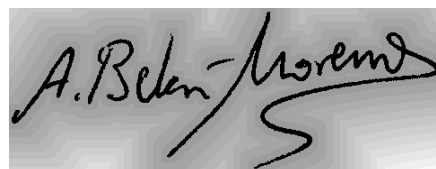


Figura 3. Ejemplo de *mapa de potencial* sobre una firma. Se ha discretizado a 16 valores y se ha aumentado el contraste para que se aprecie mejor la ilustración.

Obsérvese que en esta formulación se tiene en cuenta las energías de los puntos adyacentes a cada punto de control del *snake*. Esto se hace para evitar la inmovilidad de los extremos del *snake*.

#### Mantenimiento de forma

Con objeto de que el *snake* no pierda sensiblemente su forma original durante las sucesivas iteraciones, que se producen cuando su posición inicial no está cerca de su ajuste definitivo, se han propuesto dos términos para la energía interna:

$$E'_{forma} = E'_{angulo} + E'_{prop}$$

La primera función se encarga de mantener dentro de un rango el ángulo entre cada par de segmentos adyacentes de la línea poligonal  $P$  con respecto al ángulo que inicialmente tenían esos dos segmentos. Para ello, cuando el ángulo en una iteración se separa de ese ángulo inicial se penaliza en un valor proporcional a la desviación. Además, si el ángulo supera un umbral  $U_a$ , a partir del cual se cambia en exceso la forma, se prohíbe dicho cambio haciendo infinita la energía.

Esta función se puede expresar como:

$$E'_{\text{angulo}}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{si } \delta > U_a \\ k_a \cdot \delta & \text{si } \delta \leq U_a \end{cases}$$

donde 
$$\delta = \left| \text{ang}(\overrightarrow{v_{i-1}(0)v_i(0)}, \overrightarrow{v_i(0)v_{i+1}(0)}) - \text{ang}(\overrightarrow{v_{i-1}(t)v_i(t)}, \overrightarrow{v_i(t)v_{i+1}(t)}) \right|$$

y  $\text{ang}$  representa el ángulo formado por dos vectores y  $k_a$  es una constante que pondera la influencia del cambio del ángulo.

La segunda función tiene la misión de preservar las proporciones entre segmentos adyacentes de la línea poligonal  $P$ . Para ello se calcula el ratio  $\phi$  dado por el cociente de las proporciones iniciales de un par de segmentos adyacentes y el cociente de las proporciones del mismo par de segmentos en una iteración. La desviación respecto a 1 de este cociente  $\phi$  se penaliza hasta alcanzar cierto umbral  $U_p$  a partir del cual se prohíbe el cambio, haciendo infinita la energía. Esta función se expresa como:

$$E'_{\text{prop}}(v_{i-1}, v_i, v_{i+1}, t) = \begin{cases} \infty & \text{si } 1 - U_p \leq \phi < 1 \text{ o si } \phi > 1 + U_p \\ k_p \cdot \phi & \text{si } 1 - U_p \leq \phi \leq 1 + U_p \end{cases}$$

donde 
$$\phi = \frac{\left\| \overrightarrow{v_i(t)v_{i+1}(t)} \right\| / \left\| \overrightarrow{v_{i-1}(t)v_i(t)} \right\|}{\left\| \overrightarrow{v_i(0)v_{i+1}(0)} \right\| / \left\| \overrightarrow{v_{i-1}(0)v_i(0)} \right\|}$$

y  $k_p$  es una constante que pondera la influencia del cambio del tamaño.

Se aprecia que ambas fórmulas son compatibles con el enfoque de minimización aportado por (4).

### 4.3. Medida de la similitud

Tras un número variable de iteraciones el *snake* converge a una solución. Para estudiar la similitud entre el *snake* y la figura a la que se ha ajustado se pueden plantear dos enfoques: utilizar la función de energía proporcionada por el mismo *snake*, o utilizar otra función diferente.

El uso de la función de energía propia del *snake* tiene varios problemas. Por un lado, la función de energía sólo tiene en cuenta los puntos de control, no teniendo en cuenta la información relativa a los puntos que componen los segmentos que unen los puntos de control. Esto se hace así para acelerar el cálculo en las iteraciones, máxime teniendo en cuenta que los resultados han sido muy semejantes en las pruebas que hemos realizado computado el potencial de todos los puntos de cada segmento del *snake*.

Además, los cambios en ángulo y proporciones del *snake*, respecto a sus valores iniciales considerados en la función de energía, se encuentran limitados por los umbrales y dan poco margen para poder ser utilizados con fines de estudiar niveles de similitud.

Por último, la función de energía sólo explica cómo se ajusta el *snake* a la imagen de la firma, pero no explica cómo se ajusta la imagen de la firma al *snake*. Esto significa que la función de energía no refleja nada sobre aquellos puntos, pertenecientes a la firma a verificar, que tras las iteraciones quedan lejos del *snake*.

Todo esto obliga al uso de una función diferente para medir la similitud entre el *snake* y la firma a la que se ha ajustado. De nuevo al definir una función de similitud se puede proceder de dos formas: utilizar alguna función de *ajuste elástico* que compare el *snake* antes de las iteraciones y después, o utilizar una función que compare el *snake* tras las iteraciones con la imagen a la que se ajusta.

Experimentando con diferentes algoritmos de *ajuste elástico* [17][3] se ha comprobado que éstos no se pueden utilizar, ya que las variaciones que se producen en el *snake* a lo largo de las iteraciones son demasiado pequeñas como para que estos algoritmos den una medida fiable para la posterior clasificación.

Teniendo en cuenta todo el planteamiento precedente, se ha planteado una función propia de medida de similitud que se compone de dos partes: factor de coincidencia ( $fc$ ) y factor de distancia ( $fd$ ).

#### Factor de coincidencia

El factor definido tiene su origen en la función de energía. Al igual que aquélla, utiliza una función de *mapa de potencial*  $m_{\text{imagen}}(x,y)$  para asignar un

valor a cada punto del *snake*, pero en esta ocasión se computan todos los  $N$  puntos de los segmentos que componen el *snake*, y no sólo los puntos de control. Para obtener un valor normalizado entre 0 y 1 de la coincidencia entre el *snake* y la firma a la que se ajusta se utiliza la siguiente expresión:

$$fc = 1 - \frac{1}{N} \sum_{i=0}^N c(p(i))$$

siendo

$$c(p(i)) = \frac{m_{imagen}(p(i))}{k_{fc} / g}$$

y donde  $N$  el número de puntos considerados en el *snake*,  $p(i)$  son los puntos interiores a los segmentos del *snake*,  $g$  el grosor medio en píxeles del trazo de la firma y  $k_{fc}$  es un factor de escala.

#### Factor de distancia

Este otro factor utiliza una *mapa de potencial*  $m_{snake}(x,y)$ , que tiene su origen en el *snake* una vez iterado, y qué permite calcular a que distancia se encuentran los píxeles de la firma respecto a la posición del *snake*. La formulación de la medida de la distancia es:

$$fd = \sum_{i=0}^M d(r(i))$$

siendo

$$d(p(i)) = \begin{cases} 1 & \text{si } m_{snake}(r(i)) < g \\ 0 & \text{si } m_{snake}(r(i)) \geq g \end{cases}$$

y donde  $r(i)$  son los píxeles con valor negro de la imagen a verificar,  $M$  es el número de tales píxeles y  $g$  es el grosor medio en píxeles del trazo de la misma firma.

Esta segunda medida tiene el problema de ser sensible al ruido. Mientras que el ruido aleatorio no debe presentar mayor problema, pues se puede eliminar con un sencillo filtro morfológico, el ruido con estructura puede ser un inconveniente en caso de no eliminarse para la verificación.

## 5. Experimentos

Debido a la escasez de una base de datos de referencia, hemos creado nuestra propia base de datos de firmas (<http://gavab.escet.urjc.es>). La inexistencia de una base de datos común de referencia complica la comparación experimental de nuestro enfoque con otros existentes.

La base de datos consiste en 4 firmas tomadas a cada uno de 28 individuos. Las firmas se han tomado en instantes diferentes y utilizando variados elementos de escritura. Finalmente, han sido escaneadas como imágenes bitonales a 300

puntos por pulgada, y almacenadas en formato BMP.

La figura adjunta muestra algunas de las firmas de tal base de datos.



Figura 4. Muestra de la base de datos.

### 5.1. Resultados

Se han ensayado diferentes combinaciones para los valores de los parámetros del modelo, utilizando una muestra de aprendizaje reducida de sólo 3 individuos, dejando 25 individuos para el test. Finalmente, tras la experimentación realizada se ha optado por el uso de la combinación de parámetros expuesta en la tabla adjunta.

$U_a$	5	$k_a$	0.1
$U_p$	0.1	$k_p$	1
		$k_{fc}$	10

Tabla 1.- Valores de los parámetros usados para la verificación de firmas.

El valor de tales parámetros se puede justificar:  $U_a$  impide que el ángulo entre dos segmentos cambie en más de 5 grados;  $U_p$  impide que un segmento cambie respecto a su vecino en más de un 10%;  $k_a$  baja en un orden de magnitud el cambio de ángulo respecto a la energía de la imagen;  $k_p$  mantiene el orden de magnitud del cambio de proporciones que ya es de 0.1; finalmente  $k_{fc}$  escala los valores del factor de coincidencia para que proporcione valores en un rango entre 0 y 1.

Se ha construido un clasificador euclídeo utilizando como características discriminantes el factor de coincidencia y el factor de distancia. De nuevo, para el aprendizaje del clasificador se ha usado la muestra de aprendizaje reducida que consta de sólo 3 individuos, utilizando validación cruzada para aumentar el número de patrones.

Como se utiliza una firma de cada individuo para crear el *snake* quedan 3 firmas auténticas de cada uno de los 25 individuos de test para calcular la tasa de error al rechazar. Utilizando estos 75 casos (25x3) se ha obtenido un 8% de error al rechazar (FRR).

Para calcular la tasa de error al aceptar cada firma, se han utilizado el resto de firmas, es decir por cada firma se han realizado 96 pruebas (24x4). Como se parte de 25 individuos, esto da lugar a 2400 casos (25x96). Esto ha dado lugar a un resultado del 6,7% de error al aceptar (FAR).

Además, el número de iteraciones tras las que converge a una solución el *snake* no ha sido superior a 93 en ningún caso, empleándose un tiempo medio de 15 segundos por cada firma a verificar (en un P-IV a 3GHz sobre Windows 2000 con el algoritmo implementado en C).

## 6. Conclusiones y futuros trabajos

Se ha propuesto un algoritmo que ajusta un *snake* a una firma manuscrita. También se ha propuesto un algoritmo de verificación *off-line* de firmas en condiciones reales.

Actualmente, estamos emprendiendo trabajos en "sentido horizontal", abarcando los procesos que faltan para disponer de un sistema realmente operativo, y en "sentido vertical", mejorando los algoritmos descritos en este artículo.

En particular, los elementos que faltan para disponer de un sistema completo de verificación de firmas son dos:

- a) Un método de creación automática del *snake* a partir de la imagen de aprendizaje, para evitar su trazado manual.
- b) Un algoritmo que permita la localización y segmentación de la imagen de una firma dentro de una imagen de un documento genérico.

Por otro lado, las mejoras sobre el trabajo ya realizado se concentran en dos aspectos:

- a) La mejora del método de comparación entre el *snake* iterado y la imagen a verificar.
- b) La optimización de los algoritmos y parámetros para funcionar en los casos en que exista ruido.

## 7. Bibliografía

[1] Amir A. Amini y otros, "Using Dynamic Programming for Solving Variational Problems in Vision", IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol 12, nº 9, Sept 1990.

[2] R. Bajaj y S. Chaudhury, "Signature Verification using Multiple Neural Classifiers", Pattern Recognition, Vol. 30, no. 1, pp. 1-7, January 1997.

[3] A. Blake y M. Isard, "Active Contours", Capítulo 3, Springer 2000.

[4] J. L. Camino y otros, "Signature Classification by HMM", IEEE 33 International Carnahan Conference, Madrid 1999.

[5] N.E. Davison, "Snakes simplified", Pattern Recog., Vol 33 pags 1651-1664, 2000.

[6] B. Fang y otros, "Off-line signature verification with generated training samples", IEE Proc.-Vis. Image and Signal Processing, Vol. 149, no. 2, pp. 85-90, 2002.

[7] Z. Hou, C. Han, "Force field analysis snake: an improved parametric active contour model", Pattern Recog. Letters 26, 513-526, 2005.

[8] Investigación y Programas, S.A, informe interno.

[9] M. A. Ismail, Samia Gad, "Off-line arabic signature recognition and verification", Pattern Recog., vol 33, pp 1727-1740, 2000.

[10] E.J. Justino y otros, "The Interpersonal and Intrapersonal Variability Influences of Off-Line Signature Verification using HMM", Proc. XV Brazilian Symposium on Comp. Graphics and Image Proces. (SIBGRAPI 2002), Fortaleza (Brazil), October 2002.

[11] M. Kass y otros, "Snakes: Active Contour Models", International Journal of Computer Vision, 321-331, 1988.

[12] F. Leclerc y R. Plamondon, "Automatic Signature Verification: The State of the Art", Intl. J. Pattern Recog and Artificial Intelligence, Vol. 8, no. 3, pp. 643-660, March 1994.

[13] Y. Muzukami, y otros, "And off-line signature verification system using an extracted displacement function", Pattern Recog. Letters, Vol. 23, pp. 1569-1577, 2002.

[14] V.E. Ramesh, M. Narasimha, "Off-line signature verification using genetically optimized weighted features", Pattern Recog, vol 32, pp 217-233, 1999.

[15] J. Vélez y otros, "Robust off-line signature verification using compression networks and positional cuttings", Proc. of the IEEE Conference on NN for Signal Processing, NNSP 2003, Toulouse 2003.

[16] L. D. Cohen y I. Cohen, "Finite element method for active contour models and ballons for 2-D and 3-D images", IEEE Trans. PAMI 15 (11),1131-1147, 1993.

[17] K. Yoshida y H. Sakoe, "Online handwriting character recognition for a personal computer system", IEEE Trans. Cosum. Electron., 28(3), 202-9, CE 1982.

[18] M. Ammar y otros, "Off-line preprocessing and verification of signatures", Intl. Journal of Pattern Recog. and Arti. Intel., 1987.

[19] R. Plamondon, "Progress In Automatic Signature Verification", Series in Machine Perception a Artificial Intelligence. Vol 13, 1994.