MULTIDIMENSIONAL EARTH MOVER's DISTANCE GRADIENT FLOW SEGMENTATION

ESCUELA SUPERIOR DE INGENIEROS DE SEVILLA

Germán Bohórquez Ruiz

A Begoña Acha y Carmen Serrano por darme la oportunidad de realizar este proyecto, a Carlos S. Mendoza por hacerlo posible y a la educación pública que hizo posible todo lo demás.

Y a Sevilla, por todas las pequeñas cosas que allí encontré (y alguna otra que también perdí).

ÍNDICE:

1. INTRODUCCIÓN

1.1. PRINCIPIOS BÁSICOS DEL SEGMENTADO DE IMÁGENES

Dada la motivación que da lugar a este trabajo y el contexto en que se enmarca, en este breve capítulo introductorio pretendo dar una idea general de qué es el segmentado de imágenes y de cuáles son sus principales aplicaciones. Además se incluye una reseña histórica de cómo se originó esta rama del tratamiento de imágenes y de su posterior evolución hasta nuestros días.

La segmentación de imágenes podría definirse como el conjunto de técnicas que permiten el desarrollo e implementación de métodos y herramientas destinadas al tratamiento de imágenes con el fin de determinar y aislar determinados elementos de las mismas. Este es un campo amplio sobre el que es posible encontrar una gran cantidad de trabajos publicados y que engloba una amplia gama de líneas de investigación. El motivo de que una parte de la comunidad investigadora dedique su tiempo y esfuerzo a desentrañar los fundamentos de esta disciplina del tratamiento de imágenes deriva, sin duda, de la multitud de aplicaciones tecnológicas que están sustentadas sobre la base de estos conocimientos.

En una sociedad en la que los dispositivos y aplicaciones electrónicas son ya omnipresentes, el uso de la imagen digital se ha convertido en parte nuclear de una gran cantidad actividades, tanto de tipo puramente tecnológico como lúdico. Estas actividades abarcan una gran cantidad de áreas, como son las de las telecomunicaciones, la sanidad, la industria multimedia, la defensa, etc.

La mejor manera de ilustrar estas aplicaciones es poner algunos ejemplos significativos, en los que el segmentado de imágenes tiene una especial relevancia:

- Aplicación en la fotografía digital. En la actualidad muchas cámaras fotográficas poseen diferentes funcionalidades que permiten facilitar la captura de imágenes. Una de las funcionalidades más comunes es la de detección de rostros, la cual permite enmarcar el rostro de la persona fotografiada de manera que resulte más sencillo conseguir un encuadre y enfoque adecuado. El principio de funcionamiento de esta aplicación se basa, entre otras cosas, en la segmentación de imágenes. Básicamente el software de la máquina segmenta la imagen aislando los elementos representativos de una cara humana (ojos, boca, nariz, etc.), tras esto se compara la forma y posición relativa en la imagen de estos elementos con una base de datos de medidas biométricas con el fin de decidir si los elementos observados se corresponden con una cara humana o no.
- Aplicaciones médicas. En el campo de la medicina encontramos multitud de herramientas de diagnóstico basadas en el procesado de imágenes, todas ellas surgidas tras el impresionante desarrollo de los equipos de captación de imágenes del interior del cuerpo humano (TAC, RM, etc.). Esta área la segmentación de imágenes proporciona un contexto ideal para la aplicación del segmentado de imágenes para,

por ejemplo, detección de tumores, volumetría de tejidos y como paso previo necesario para aplicar otros procesos de tratamiento de imágenes.

En las siguientes páginas daremos una idea general del estado actual de las técnicas de segmentación de imágenes y presentaremos el algorítmico que motiva la elaboración del presente trabajo.

2. MULTIDIMIENSIONAL EARTH MOVER'S DISTANCE GRADIENT FLOW SEGMENTATION

2.1. IMAGE SEGMENTATION: THE STATE OF THE ART

Inicialmente haremos un breve resumen de las principales técnicas de segmentado de imágenes con la intención de ubicar el algoritmo propuesto dentro del amplio sistema de técnicas de segmentado desarrolladas en la actualidad.

En función de la literatura consultada es posible establecer diferentes criterios a la hora de clasificar las diferentes técnicas de segmentado de imágenes. En este caso nos decantamos por utilizar el criterio más general, el cual es ampliamente aceptado por la mayoría de los autores. Este criterio divide, inicialmente, los algoritmos clásicos de segmentado es dos grandes grupos: Técnicas de segmentado basadas en regiones (region based) y técnicas de segmentado basadas en contornos (edge based). Adicionalmente existiría un tercer grupo de técnicas denominadas de umbralización; las más usuales están basadas en el estudio del histograma de la imagen y consisten en la determinación de uno o varios umbrales tales que determinen los distintos rangos de intensidad que definen los objetos presentes en la imagen, por ser técnicas bastante rudimentarias y de aplicación limitada no se profundizará en el análisis y descripción de las mismas. Respecto a los dos grupos anteriormente citados, a continuación intentaremos mostrar una visión global de los mismos.

- Region Based: Los métodos de segmentación basados en regiones funcionan de tal forma que dados unos pixeles "semilla" (seeds), se realiza un proceso iterativo por el cual los píxeles vecinos a las "semillas" se incluyen en la región objetivo o no en función de una regla de decisión conocida generalmente como criterio de homogeneidad, este proceso se lleva a cabo tantas veces como sea necesario hasta que ya no haya más píxeles para añadir. Este método es conocido como crecimiento de regiones (*region growing*). Usando una visión complementaria, existe una versión distinta de este algoritmo, conocida como división de regiones (*region splitting*) por la cual se toma el total de la imagen y se analizan los píxeles de la misma con el fin de verificar si todos ellos cumplen con alguna condición de similitud, si se cumple esta condición se considerará que toda la región se corresponde con la región objetivo, en caso contrario se subdivide la imagen y se considera cada una de ellas a su vez como el área de interés. Este proceso se repite hasta que no se produzca ninguna nueva división de regiones. Existen también técnicas mixtas en las que se combinan ambos enfoques (*splitting & merging*).

  Region growing/splitting: De forma esquemática podemos definir el proceso como sigue

  1. $I = \bigcup_{i}^{n} R_i$ (I=Imagen)

2. $R_i \bigcap R_j = \varnothing; \forall i \neq j$
3. $H(R_i) = \text{true}; \forall i$
4. $H(R_i \bigcup R_j) = \text{false}; \forall R_i, R_j \text{vecinos}, i \neq j$

Donde $H(R_i)$ mide la homogeneidad de la región i. En un proceso de region growing se parte de las diferentes semillas y se van añadiendo los píxeles que cumplen la condición y en uno de region splitting se parte del total de la imagen y se va dividiendo está hasta que todas las subregiones generadas verifiquen el criterio de homogeneidad. Los principales esfuerzos en esta área se centran en el estudio y generación de criterios de homogeneidad que permitan realizar segmentaciones certeras y eficientes para una tipología de imágenes concreta o de propósito general.

- Edge Based: Esta otra metodología se basa en la evolución de un contorno de forma que este se ajuste a la región objetivo de la imagen. Matemáticamente esto se traduce en la resolución, analítica o numérica, de una función en derivadas parciales (Partial Differential Equation: PDE). Por ello estos métodos también son conocidos como PDE-Based. La resolución de la PDE implica necesariamente unas restricciones y condiciones de regularidad que se ven reflejada en la geometría de la curva. Los métodos de segmentado basados en contornos pueden a su vez dividirse en tres categorías básicas.

    - Métodos paramétricos: Conocidos también como técnicas de Lagrange (o *snakes*), se basan en la parametrización del contorno de acuerdo con alguna estrategia de muestreo y posteriormente una evolución de cada elemento del contorno de acuerdo a determinadas características de la imagen.

    - Métodos *Level-Set*: Estas técnicas se basan en el uso de una función "de nivel", donde el nivel cero se corresponde con el contorno que se pretende ajustar a la región objetivo de la segmentación. En base esto y definiendo una función de velocidad se hace evolucionar este contorno. Existen dos formas clásicas de resolver la evolución de contornos, una consiste en actualizar los valores de la curva sobre toda la imagen (i.e. Chan-Vese) y otra en la que únicamente se resuelve la PDE sobre un entorno reducido conformado por los píxeles vecinos al contorno (narrow band).

    - Métodos Fast Marching: Es un caso especial del método level-set y se basa en la resolución numérica de problema de los contornos de la ecuación de Eikonal [$F(x)|\nabla T(x)| = 1$]. Típicamente esta ecuación describe la evolución en un intervalo de tiempo T de una curva bajo la influencia de un campo de fuerzas F(X) (velocidad) en la dirección normal a un punto x de dicha curva.

La técnica de segmentación que presentamos en este trabajo se encuadraría en la categoría Level-Set (Narrow Band). Una vez definido el método genérico de segmentación procederemos a describir la función de velocidad que usamos.

2.2. EARTH MOVER's DISTANCE

Centrándonos en los métodos level-set, planteamos a continuación la lógica que rige el funcionamiento de estos modelos. Estas técnicas se basan la evolución de una curva (ϕ), de forma que esta se ajuste al contorno de la región objetivo. Las fuerzas que modelan el movimiento de esta curva actúan de forma que la disimilitud entre las áreas englobada dentro y fuera de ϕ sea máxima, esto es lo que en el presente trabajo, entre otros, se denomina *Maximal Discrepancy*. Alternativamente se plantea otra forma diferente de modelar la evolución de la curva de forma que la disimilitud del área englobada dentro de esta sea mínima en comparación con el área englobada por otra curva denominada región modelo o *template*, esta forma de operación se conoce como *Match to Template*.

Una vez planteado esto, inmediatamente surge la necesidad de encontrar una forma que permita medir la disimilitud entre dos áreas de una imagen. Una forma habitual de medir esto es la comparación entre los histogramas de estas áreas.

Para ello, en primer lugar, será necesario definir que es un histograma en el contexto de tratamiento de imágenes. Sea X, una medida que puede tomar cualquiera de los *T* posibles valores pertenecientes al conjunto finito $X = \{x_1, x_2, \ldots, x_T\}$, si consideramos un conjunto de *n* elementos que representan el resultado de las mediciones realizadas sobre X, $A = \{a_1, a_2, \ldots, a_n\}$, at $\in$ X. Entonces el histograma del conjunto A sobre las medidas de X, denotado como H(X,A), el cual representa de forma ordenada, el número de ocurrencias de los valores discretos de X sobre a_t. Si estamos interesados únicamente en medir la disimilitud entre dos histogramas, todos ellos sobre la misma medida X. sueles asimilarse sin pérdida de generalidad H(X,A) ≡ H(A). Entonces $H_i(A)$, con $1 \leq i \leq T$, representa el número de elementos de A que toman el valor x_i, es decir, $H(A) = [H_1(A), H_2(A), \ldots, H_T(A)]$. Donde

$$H_i(A) \sum_{i=1}^{n} C_{i,T}^{A}$$

Definiéndose la función de coste individual como: $C_{i,T}^{A} = 1$ si $a_t = x_i$, 0 e.o.c. Los elementos de $H_i(A)$ son lo que se conocen habitualmente como *bins*.

A lo largo del tiempo se han propuesto multitud de formas para medir la disimilitud entre dos histogramas. Habitualmente los métodos de medida se han clasificado en dos grandes categorías: bin-to-bin y cross-bins. Los métodos tipo bin-to-bin se basan en comparar únicamente aquellos bins con correspondencia 1 a 1 de cada una de las distribuciones. Sean $H_i(A)$ y $K_j(A)$ los dos histogramas de los cuales pretendemos medir su disimilitud, entonces en una medida tipo bin-to-bin se comparan h_i y k_j $\forall$ i = j, omitiendo todos aquellos casos en los que $i \neq j$. Los métodos tipo cross-bin incluyen en la comparación también los términos asociados a los bins no correspondientes. Para ello en una medida cross-bin se hace uso de la distancia (*ground distance*) entre bins no correspondientes [d_{ij}], definida como la distancia (medida en función de alguna cualidad representativa de los bins) entre el bin *i* y el *j*. A continuación describiremos brevemente algunos de los métodos clásicos de medida de la disimilitud entre histogramas:

Bin-to-Bin: Como ya hemos dicho, en esta categoría solo los bins de los histogramas con el mismo índice son comparados. Esto puede interpretarse como el uso de una ground distance, en el caso particular de que esta es una distancia binaria con un umbral dependiente del tamaño del bin, esto es debido a que valores similares se engloban dentro del mismo bin. El umbral utilizado para incluir o no elementos dentro del bin constituye el tamaño del mismo. Para el caso bin-to-bin las principales medidas de disimilitud clásicas son:

- *Minkowsky-Form distance*:

$$d_{Lr}^{(H,K)} = \left(\sum_i |h_i - k_i|^r\right)^{1/r}$$

  Siendo las más conocidas las distancias L_1, L_2 y L_∞.
- *Intersección de histogramas*: En este caso cuando área de ambos histogramas es la misma esta medida se corresponde con las distancia L_1 normalizada.

$$d_{\cap}(H,K) = 1 - \frac{\sum_i \min(h_i - k_i)}{\sum_i k_i}$$

- *Kullback&Leibler Divergence (KLD) y Joffrey Divergence (JD)*: La KLD se define como

$$d_{KLD}(H,K) = \sum_i h_i \cdot \log\left(h_i / k_i\right)$$

  Para corregir ciertas propiedades no deseadas de esta medida, como la simetría o la dependencia con la elección de los bins, se define una nueva medida basada en la anterior, Joffrey Divergence:

$$d_{JD}(H,K) = \sum_i \left(h_i \cdot \log\left(h_i / m_i\right) + k_i \cdot \log\left(k_i / m_i\right)\right)$$

  Donde $m_i = (h_i + k_i) / 2$.
- *Medida estadística χ^2*: en este caso la medida se define como

$$d_{\chi^2}(H,K) = \sum_i (h_i - m_i)^2 / m_i$$

  Donde $m_i = (h_i + k_i) / 2$.

Cross-bin: El mayor inconveniente de las medidas bin-to-bin es su alta dependencia del tamaño del bin, esto implica que para tamaños pequeños no poseen de un nivel de discriminación suficiente en la mayoría de los casos. Cuando la distancia se asemeja con una distancia perceptual podemos incorporar esta información para medir la disimilitud entre dos histogramas otorgándoles una función diferenciadora desde el punto de vista de la percepción de determinadas características de la imagen. Algunas de las medidas cross-bin más conocidas son:

- Quadratic Form Distance (Niblack, 1993):

$$d_Q(H,K) = \sqrt{(\bar{h} - \bar{k})^T \cdot A (h - k)}$$

  Donde **h** y **k** son vectores los cuales incluyen todos los elementos que conforman H y K. La información cross-bin se introduce mediante la matriz de similitud $A = [a_{ij}]$,

en la que cada elemento a_{ij} indica la similitud existente entre el bin *i* y el *j*. Niblack recomienda usar $a_{ij} = 1 - d_{ij} / d_{max}$. Siendo d*ij* la distancia entre los bin *i* y *j*.

- *Match Distance*:

$$d_{MD}(H,K) = \sum_i |\hat{h}_i - \hat{k}_i|$$

Donde $\hat{h}_i = \sum_{i \le j} h_j$ y $\hat{k}_i = \sum_{i \le j} k_j$ son los valores acumulados de los histogramas H y K.

- *Kolmogorov-Smirnov distance*:

$$d_{KS}(H,K) = \max_i(|\hat{h}_i| - |\hat{k}_j|).$$

Las anteriores medidas de distancia tan solo son posibles de aplicar sobre distribuciones de una dimensión ya que para distribuciones de orden superior no es posible realizar una ordenación unívoca de los elementos del histograma.

Habitualmente, para un conjunto de medidas, únicamente una pequeña parte de los bins presentes en el histograma contienen información significativa y muchos de los bins están vacíos. En estos casos una estructura basada en un tamaño fijo de los bins es poco eficiente; por este motivo Rubner propuso una descripción de los bins de tamaño variable a la que denominó signatura. En este tipo de representaciones los bins vacíos no están explícitamente representados. Formalmente definimos la signatura de una imagen como sigue: dados $H(A) = [H_1(A), H_2(A), \ldots, H_T(A)]$ y $S(A) = [S_1(A), S_2(A), \ldots, S_T(A)]$, el histograma y la signatura del conjunto A respectivamente. Cada $S_k(A)$, $1 \le k \le z \le T$, está compuesto por un par de términos $S_k(A) = \{w_k, m_k\}$. El primer término, w_k, representa la relación entre la signatura S(A) y el histograma H(A). Si $w_k = i$, entonces el segundo término, m_k, es el número de elementos de A que toman el valor x_i, esto es, $m_k = H_i(A)$ donde $w_k < w_t \leftrightarrow k < t$ y $m_k > 0$. LA signatura del conjunto es por tanto una representación del histograma en la cual los bins vacíos no son explícitamente considerados. De la definición de signatura de deriva que: $H_{wk}(A) = m_k$, donde $1 \le k \le z$.

Las principales medidas de disimilitud entre signaturas son:

- *Distancia nominal*: $d_{nom}(a, b) = 1$ si $a = b$; 0 e.o.c.
- *Distancia ordinal*: $d_{ord}(a, b) = |a-b|$.
- *Distancia módulo*: $d_{mod}(a, b) = |a-b|$ si $|a-b| \le T/d$; $T-|a-b|$ e.o.c.

A continuación presentamos el algoritmo de Earth Mover's Distance (EMD), interpretándolo como una métrica para la medida de disimilitud entre dos signaturas. El "problema del transporte", también conocido como *consumers-suppliers problem*, es un problema matemático ampliamente conocido y estudiado. El cual se plantea en el siguiente escenario. Un conjunto finito de proveedores, cada uno de ellos con un volumen de bienes para comerciar conocido y limitado, ha de abastecer a otro conjunto de consumidores también con unas necesidades de consumo conocidas y limitadas. Para cada par *consumer-supplier* el coste de transportar una unidad del bien comercializado es igualmente conocido. En este escenario la solución al problema se fundamentaría en encontrar los flujos de transporte de bienes *supplier* $\rightarrow$ *consumer* que permitan abastecer toda la demanda a un coste mínimo.

La transformación (comparación) de una distribución (signatura) de una imagen en otra objetivo puede interpretarse en términos de este problema y resolverse usando la formulación de propuesta por Monge-Kantorovich, los cuales plantearon un esquema formal de resolución basado en métodos de programación lineal. Para ello, dada una imagen, representada por su signatura:

$$P = \{(p_1,w_1), (p_2,w_2), \dots ,(p_i,w_i), \dots , (p_m,w_m)\} \equiv \text{Signatura de } m \text{ clusters.}$$

Donde p_i representa las coordenadas del centroide del cluster i y w_i el peso de dicho cluster. Análogamente, podemos definir para la imagen objetivo la siguiente signatura:

$$Q = \{(q_1,w_1), (q_2,w_2), \dots ,(q_j,w_j), \dots , (q_n,w_n)\} \equiv \text{Signatura de } n \text{ clusters.}$$

Entonces, definida la matriz de distancias $D = [d_{ij}]$, conformada por los distintos $d_{ij} \equiv$ distancia entre el cluster p_i y q_j y la matriz de flujos $F = [f_{ij}]$, conformada por los elementos $f_{ij} \equiv$ flujo de transporte entre el cluster p_i al q_j; la función de coste a optimizar sería:

$$W(P,Q,F) = \sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} \bullet f_{ij}$$

Sujeta a las siguientes restricciones:

C1: $f_{ij} \geq 0 \qquad 1 \leq i \leq m \qquad 1 \leq j \leq n$

C2: $\sum_{i=1}^{m} f_{ij} \leq w_{pi} \qquad 1 \leq i \leq m$

C3: $\sum_{j=1}^{n} f_{ij} \leq w_{qi} \qquad 1 \leq j \leq n$

C4: $\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij} = \min(\sum_{i=1}^{m} w_{pi}, \sum_{j=1}^{n} w_{qj})$

La restricción *C1* implica que los flujos de bienes han de trasportarse desde los proveedores (P) hasta los consumidores (Q) y no al revés. Por su parte, las restricciones *C2* y *C3* establecen que el flujo de bienes desde un cluster de P (proveedor) a otro de Q (consumer), ni el peso del cluster origen (en P) ni el del cluster destino (en Q); en otras palabras, ni un proveedor poder servir más bienes de los que dispone, ni un consumidor recibir más de los que demanda. Por último, la restricción *C4* fuerza a que la transferencia de bienes sea máxima.

Con la resolución del anteriormente descrito problema del transporte hallando con ello el flujo óptimo (F), podemos definir la EMD como el trabajo total realizado, normalizado por el flujo total:

$$EMD(P,Q) = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n} d_{ij} \bullet f_{ij}}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{ij}}$$

Una gran ventaja derivada del uso de esta métrica (EMD) es que la resolución del problema de transporte puede ser eficientemente implementada usando el método SIMPLEX de Hillier & Lieberman.

En el presente trabajo se propone usar esta EMD como medida de la disimilitud entre dos signaturas. Para ello se hace necesario derivar una función de fuerza que permita modelar la evolución de contornos para el algoritmo de segmentación de imágenes propuesto.

2.3. EL MÉTODO SIMPLEX Y ANÁLISIS DE SENSIBILIDAD EMD

El algoritmo anterior puede expresarse usando una formulación matricial, la cual permite realizar de forma más intuitiva el análisis de sensibilidad del mismo. Así la matriz de inicio puede transformarse en la matriz óptima, usando el algoritmo SIMPLEX, variando la EMD en función del peso de los clusters. Sean N^2 variables f_{uv} y N^2 constantes d_{uv}, representadas en los vectores (1xN^2) **f** y **d**, respectivamente; los cuales contienen los conjuntos de valores asignados a los flujos y a las distancias:

$$\vec{f} = [f_{11} \ldots f_{1N} \ldots f_{N1} \ldots f_{NN}]^T$$

$$\vec{d} = [d_{11} \ldots d_{1N} \ldots d_{N1} \ldots d_{NN}]^T$$

A su vez las restricciones expuestas en el punto anterior pueden expresarse matriz de coeficientes c_{uv} (2N+1 x N^2), con $1 \le u \le N$, $1 \le v \le N$; donde los subíndices *uv* de los coeficientes indican que el flujo f_{uv} aparece en la correspondiente restricción.

$$H = \begin{bmatrix} c_{11} & 0 & \ldots & 0 & \ldots & c_{N1} & 0 & \ldots & 0 \\ & & & & \ldots & & & & \\ 0 & \ldots & 0 & c_{1N} & \ldots & 0 & \ldots & 0 & c_{NN} \\ c_{11} & \ldots & \ldots & c_{1N} & \ldots & 0 & \ldots & \ldots & 0 \\ & & & & \ldots & & & & \\ 0 & \ldots & \ldots & 0 & \ldots & c_{N1} & \ldots & \ldots & c_{NN} \\ c_{11} & \ldots & \ldots & c_{1N} & \ldots & c_{N1} & \ldots & \ldots & c_{NN} \end{bmatrix}$$

Adicionalmente se define el vector **b** (1 x 2N), el cual contiene los pesos como:

$$\vec{b} = [w_1^O \ldots w_N^O w_1^I \ldots w_N^I]^T$$

Entonces es posible reformular $\arg\min_{f_{uv}} [Z(f_{uv}, d_{uv}) = \sum_{u=1}^{N} \sum_{v=1}^{N} d_{uv} f_{uv}]$ = EMD (s^I,s^O) como:

$$\arg\min_{\vec{f}} Z = \vec{d}^T \vec{f}$$

Sujeto a las siguientes restricciones: $H\vec{f} = \vec{b}$ y $\vec{f} \ge \vec{0}$.

Tal como establece el algoritmo SIMPLEX, si existen N^2 variables y 2N + 1 restricciones siempre es posible dividir el conjunto de variables en dos grupos, uno de ellos de 2N + 1 variables básicas y otro de N^2 – (2N + 1) variables no básicas. En base a esto se pueden reescribir los elementos que forman parte del problema como la unión de dos componentes, una asociada a las variables básicas (B) y otra asociada a las variables no básicas (NB):

$$\vec{f} \to [\vec{f}_B^T \mid \vec{f}_{NB}^T]^T \; ; \; \vec{d} \to [\vec{d}_B^T \mid \vec{d}_{NB}^T]^T \; ; \; H \to [H_B \mid H_{NB}]$$

Centrándonos en el criterio de máxima discrepancia, para un contorno dado se establece una energía (E) inversamente proporcional a la EMD entre las signaturas de las porciones de imagen que quedan dentro y fuera de dicho contorno, es decir, E = -EMD(s^I, s^O). En base a esto podemos definir que minimizar esta energía, E, puede expresarse de forma equivalente como el siguiente problema de optimización: argmax $_{[\phi(\mathbf{x})]}$ EMD.

En base al sistema de ecuaciones y restricciones anterior podemos reformular el problema de forma que $Z = \vec{d}_B^T H_B^{-1} \vec{b}$; en este contexto si sustituimos $\vec{b}$ por $\vec{b}'$, siendo $\vec{b}' = \vec{b} + [0...0 \triangle w_i^{I/O} 0...0]^T$ (donde se representa que uno de los pesos cambia manteniéndose el resto constante). Entones podemos escribir:

$$Z' = \vec{d}_B^T H_B^{-1} \vec{b}' = \vec{d}_B^T H_B^{-1} \vec{b} + \vec{d}_B^T H_B^{-1} [0...0 \triangle w_i^{I/O} 0...0]^T = \vec{d}_B^T H_B^{-1} \vec{b} + k_i \triangle w_i^{I/O}$$

Donde: $k_i = \sum_{l=1}^{2N} (\vec{d}_B)_l (H_B^{-1})_{li}; i = 1,...,2N$. Entonces podemos expresar la variación de Z respecto a los pesos como:

$$\frac{\partial Z}{\partial w_v^O} = \lim_{\triangle w_v^O \to 0} \frac{\Delta Z}{\Delta w_v^O} = \frac{k_v \Delta w_v^O}{\Delta w_v^O} = k_v$$

$$\frac{\partial Z}{\partial w_u^I} = \lim_{\triangle w_u^I \to 0} \frac{\Delta Z}{\Delta w_u^I} = \frac{k_u \Delta w_u^I}{\Delta w_u^I} = k_u$$

Con u, v = 1,…,N. Si a esto incorporamos que el cambio en el peso de un cluster ha de afectar al peso del resto de clusters pertenecientes a la misma signatura, tenemos que:

$$\frac{\partial Z}{\partial w_v^O} = k_v - \sum_{j \neq v} k_j \frac{w_j^O}{\sum_{j \neq v} w_j^O}$$

$$\frac{\partial Z}{\partial w_u^I} = k_{u+N} - \sum_{k \neq u} k_{k+N} \frac{w_k^I}{\sum_{k \neq u} w_k^I}$$

Donde: v = j = 1,…,N; u = k = 1,…,N.

2.4. MULTIDIMENSIONAL EMD GRADIENT FLOW

Como se explicaba en el punto anterior es habitual usar en el segmentado de imágenes el criterio de máxima discrepancia, por el cual dado un contorno se establece una energía, E, la cual es inversamente proporcional a la EMD entre la parte de la imagen interior (I) y exterior (O) a dicho contorno; $E = -EMD(S^I,S^O)$. Y por tanto es posible plantear la minimización de esta energía en términos del siguiente problema de optimización:

$$\underset{\Phi(\vec{x})}{\arg\max}(EMD)$$

Donde el contorno estaría contenido en el nivel cero de la función $\Phi(\vec{x})$. Para un par de signaturas dadas es posible modelar la variación de la EMD respecto al peso de las signaturas mediante la función a optimizar Z usando la variación de esta respecto a la variación de los pesos; esto se expresa en términos de un problema de optimización lineal como sigue:

$$\underset{\Phi(\vec{x})}{\arg\max}[\underset{f_{uv}(\Phi(\vec{x}))}{\arg\min}(Z(f_{uv}(\Phi(\vec{x})),d_{uv})]$$

En el cual la derivada de la EMD respecto a los pesos se puede aproximar en términos de un problema de programación lineal (usando, en este caso, el método SIMPLEX).

$$\frac{\partial EMD}{\partial w_u^I} \simeq \frac{\partial Z_{min}}{\partial w_u^I} \qquad \frac{\partial EMD}{\partial w_v^O} \simeq \frac{\partial Z_{min}}{\partial w_v^O}$$

Entonces, aplicando la regla de la cadena, podemos transformar estas variaciones (en términos de primera derivada) de la EMD respecto a ϕ como:

$$\frac{\delta EMD}{\delta \Phi} = \sum_{u=1}^{N} \frac{\partial EMD}{\partial w_u^I} \frac{\delta w_u^I}{\delta \Phi} + \sum_{u=1}^{N} \frac{\partial EMD}{\partial w_v^O} \frac{\delta w_v^O}{\delta \Phi}$$

Sea H(·) la función de Heaviside y $C_i(\mathbf{x})$ la función que indica la pertenencia o no de **x** en el cluster i {$C_i(\mathbf{x})$ = 1 si $\mathbf{x} \in s_i$, 0 e.o.c.}, podemos expresar el peso de las signaturas en función de ϕ(**x**), teniendo en cuenta que ϕ(**x**) es positivo si está dentro del contorno y negativo en caso contrario, como:

$$w_u^I(\phi(\vec{x})) = \frac{\int_{\Omega} C_u(\vec{x}) H(\phi(\vec{x}))d\vec{x}}{\int_{\Omega} H(\phi(\vec{x}))d\vec{x}}$$

$$w_v^O(\phi(\vec{x})) = \frac{\int_{\Omega} C_v(\vec{x})(1 - H(\phi(\vec{x})))d\vec{x}}{\int_{\Omega} (1 - H(\phi(\vec{x})))d\vec{x}}$$

Por tanto, siguiendo lo descrito en el análisis de sensibilidad anterior, podemos obtener la derivada de los pesos respecto a ϕ como sigue:

$$\frac{\delta w_u^I}{\delta \phi} = -\frac{\delta(\phi)}{A_I}(w_u^I - C_u(\vec{x}))$$

$$\frac{\delta w_v^O}{\delta \phi} = \frac{\delta(\phi)}{A_o}(w_v^O - C_v(\vec{x}))$$

Siendo en este caso A_I y A_O las áreas de las regiones que quedan, respectivamente, dentro y fuera del contorno. Finalmente la evolución del contorno minimizando la energía puede expresarse como la primera variación de *E* respecto a ϕ como sigue:

$$\phi_t = -\frac{\delta E}{\delta \phi} = \frac{\delta EMD}{\delta \phi} = \frac{\delta(\phi)}{A_O}\sum_{v=1}^{N}(w_v^O - C_v(\vec{x}))\frac{\delta EMD}{\delta w_v^O} - \frac{\delta(\phi)}{A_I}\sum_{u=1}^{N}(w_u^I - C_u(\vec{x}))\frac{\delta EMD}{\delta w_u^I}$$

Surge la necesidad de aplicar algún criterio de regularización sobre $\delta(\phi)$, por diferentes motivos el criterio de regularización usado es $|\nabla\phi|$. Esta regularización nos permite expresar la evolución del contorno en los siguientes términos:

$$\frac{\delta C}{\delta t} = F\vec{N}$$

Siendo F,

$$F = \frac{1}{A_O}\sum_{v=1}^{N}(w_v^O - C_v(\vec{x}))\frac{\delta EMD}{\delta w_v^O} - \frac{1}{A_I}\sum_{u=1}^{N}(w_u^I - C_u(\vec{x}))\frac{\delta EMD}{\delta w_u^I}$$

Y $\vec{N}$, el vector normal al contorno. Esta fuerza nos permite minimizar la energía *E* al tiempo que es resuelto el problema de programación lineal, para cada $t = t_0 + \Delta t$, generado por la ecuación discreta del gradiente descendente en la que en cada iteración se actualizan los valores de $\frac{\partial EMD}{\partial w_u^I}$ y $\frac{\partial EMD}{\partial w_v^O}$ tal y como se ha visto anteriormente.

Desde el punto de vista de la estimación estadística, la función de coste definida, puede presentar una alta sensibilidad ante el ruido de la imagen o errores en los datos. Por ello es necesario atenuar o eliminar las componentes espectrales debidas al ruido. Una forma de hacer esto consiste en limitar la longitud del contorno activo, en ese caso la función de energía se expresaría como:

$$E = -EMD(S^I, S^O) + \mu\, Length\,(C) \qquad \text{con } \mu > 0$$

La constante de regularización (μ) indica el compromiso entre la fidelidad del contorno y su estabilidad. Entonces podemos expresar el problema de linearización como:

$$\underset{\phi(\vec{x})}{\arg\min} = (-EMD(\phi) + \mu\int_{\Omega}|\nabla H(\phi)|\, d\vec{x})$$

Por tanto el gradiente que permite minimizar la función de coste, puede expresarse, en concordancia con lo visto anteriormente, como:

$$\phi_t = \frac{\delta EMD}{\delta \phi} - \mu \kappa \delta(\phi)$$

Siendo κ la curvatura del contorno, la cual puede expresarse de la siguiente manera:

$$\kappa = \nabla\left(\frac{\nabla \phi}{|\nabla \phi|}\right)$$

Finalmente, esto es equivalente a la evolución del contorno en la dirección normal, de acurdo con la fuerza, *F*, definida anteriormente

$$F_{reg} = F - \mu K$$

Con el desarrollo hecho en el presente capítulo hemos conseguido definir el cuerpo de ecuaciones que rigen el funcionamiento del algoritmo de segmentado. A continuación procederemos a trasladar este conjunto de razonamientos y ecuaciones al campo de la computación, plasmando los pasos dados para transcodificar el algoritmo al lenguaje computacional. Así mismo se presentarán los diferentes modos de funcionamiento y opciones de ejecución que han sido diseñados, además de mostrar de forma empírica el funcionamiento del algoritmo.

3. DESCRIPCIÓN DEL ALGORITMO

3.1 MAXIMAL DISCREPANCY OPERATION MODE

En esta sección procederemos a dar una visión detallada del funcionamiento del algoritmo, aunque en una primera aproximación tan solo describiremos el funcionamiento del modo de operación Maximal Discrepancy.

En primer lugar procesamos la imagen de entrada, la imagen que queremos segmentar, de forma que la convertiremos al espacio de color L*a*b* usando las siguientes relaciones:

$$L^\star = 116 f(Y/Y_n) - 16$$
$$a^\star = 500\,[f(X/X_n) - f(Y/Y_n)]$$
$$b^\star = 200\,[f(Y/Y_n) - f(Z/Z_n)]$$

donde:

$$f(t) = \begin{cases} t^{1/3} & \text{para } t > (\frac{6}{29})^3 \\ \frac{1}{3}\left(\frac{29}{6}\right)^2 t + \frac{4}{29} & \text{para otro valor} \end{cases}$$

Los parámetros X_n, Y_n y Z_n son los valores del triestímulo CIE XYZ del punto blanco de referencia, en este caso usamos el iluminante estándar D50.

Una vez convertida la imagen obtenemos una matriz I_{LAB} (MxNx3) sobre la cual aplicaremos el algoritmo de segmentación de imagen, procedemos a adquirir el contorno inicial el cual puede obtenerse, en el modo de funcionamiento maximal discrepancy, de cuatro formas diferentes:

- Contorno inicial rectangular (definido por el usuario)
- Contorno inicial de forma arbitraria (definido por el usuario)
- Contorno inicial conformado por un mallado circular (densidad del mallado definida por el usuario)
- Contorno inicial obtenido mediante autoinicialización

A continuación mostramos un ejemplo de cada uno de estos tipos de inicialización:

Fig. 1.A (Rectangular)

Fig. 1.B (Arbitraria)

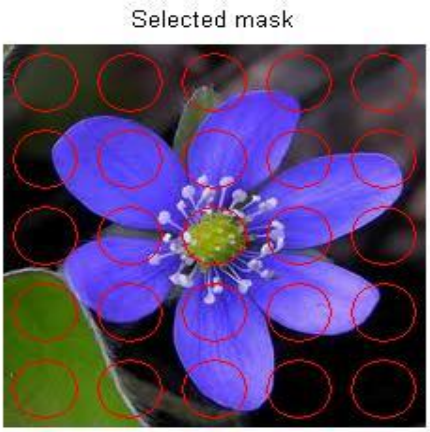


Fig. 1.C (Mallado Circular)

Fig. 1.D (Autoinicialización)

Una vez hemos obtenido el contorno inicial, la ejecución del algoritmo continua con la definición de los distintos clústeres y sus respectivos centroides. Para llevar a cabo esta división de la imagen usamos un algoritmo implementado en Matlab denominado kmeans. El modo de funcionamiento de este algoritmo es el tal que utiliza secuencialmente dos algoritmos iterativos para minimizar la suma de distancias punto a centroide, en cada uno de los k clusters:

1. La primera fase utiliza *actualizaciones por lotes*, donde cada iteración consiste en la reasignación de puntos a su centro de gravedad más cercano, seguido por un nuevo cálculo de los centroides de grupo. Esta fase, eventualmente, no converge a una solución que sea un mínimo local, es decir, una partición de los datos tal que el la reubicación de cualquier punto aumenta la suma total de las distancias punto-centroide del sistema. Esto es más probable para los conjuntos de datos pequeños. Normalmente, la segunda fase del algoritmo es la que consigue alcanzar el mínimo local.

2. La segunda fase utiliza *actualizaciones en punto a punto*, donde los puntos son reasignados individualmente, si con ello se reduce la suma de las distancias se vuelven a calcular los centroides de grupo después de cada cambio de destino. Cada iteración en la

segunda fase consiste en una única reasignación y recalculo de las distancias para todos los puntos. La segunda fase convergerá en un mínimo local, aunque puede haber otros mínimos locales con menor suma de las distancias, el problema de encontrar el mínimo global sólo puede ser resuelto, en general, por una exhaustiva elección de los puntos de partida, pero el uso de varias repeticiones, con un punto de partida aleatorio generalmente resulta en una solución que es un mínimo local.

Una vez que hemos ejecutado el algoritmo *kmeans* obtenemos un vector con las coordenadas de los centroides en el espacio de color L*a*b y una matriz con las etiquetas que asocian cada punto de la imagen a uno de los distintos clusters.

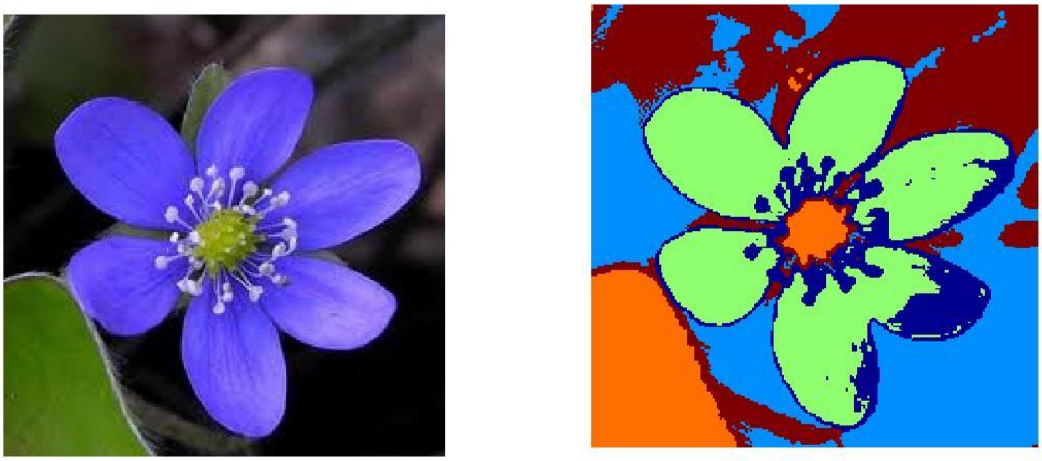


Fig. 2. Imagen original y representación de los distintos clusters [INDEXs] con M = 5.

$$\text{kmeans}\ (\text{Image}_{LAB}) \rightarrow [C] = \begin{matrix} L_1 & a_1 & b_1 \\ L_2 & a_2 & b_2 \\ \cdots & \cdots & \cdots \\ L_M & a_M & b_M \end{matrix} \quad [\text{INDEXs}] = \begin{matrix} p_{11} & p_{12} & \cdots & p_{1m} \\ p_{21} & p_{22} & & \cdots \\ \cdots & & \cdots & \\ p_{n1} & \cdots & & p_{nm} \end{matrix}$$

El siguiente paso es calcular la distancia CIE96 entre cada uno de los centroides y el resto, la distancia CIE96 entre dos puntos (del espacio L*a*b, en este caso) se calcula de la siguiente forma. Dados dos puntos X e Y pertenecientes al espacio de color L*a*b ($\{X_L, X_a, X_b\}$ y $\{Y_L, Y_a, Y_b\}$) la distancia CIE96 entre ellos, CIE96(X, Y), se obtiene siguiendo los siguientes pasos:

1. Conversión espacio LCh:

$$L_{LCh} = L_{LAB} \rightarrow \Delta L_{LCh} = L_{LAB\_X} - L_{LAB\_Y}$$

$$C_{LCh} = (a_{LAB}^2 + b_{LAB}^2)^{1/2} \rightarrow \Delta C_{LCh} = C_{LAB\_X} - C_{LAB\_Y}$$

$$h_{LCh} = \arctan(b_{LAB}/a_{LAB}) \rightarrow \Delta h_{LCh} = h_{LAB\_X} - h_{LAB\_Y}$$

2. Cálculo de la distancia en el espacio LCh:

$$\Delta E_{96} = \sqrt{\left(\frac{\Delta L_{LCh}}{S_L}\right)^2 + \left(\frac{\Delta C_{LCh}}{S_c}\right)^2 + \left(\frac{\Delta h_{LCh}}{S_h}\right)^2 + R_T \frac{\Delta C_{LCh}}{S_C} \frac{\Delta h_{LCh}}{S_h}}$$

Donde S_L, S_C, S_h y R_T son valores constantes. Finalmente obtenemos la matriz $[d]_{MxM}$ en la formada por los elementos d_{ij} := distancia CIE96 entre el centroide i y el j, $\forall\ i \neq j$ ($d_{ii} = 0$).

$$[C] = \begin{matrix} L_1 & a_1 & b_1 \\ L_2 & a_2 & b_2 \\ \dots & \dots & \dots \\ L_M & a_M & b_M \end{matrix} \rightarrow [d] = \begin{matrix} d_{11} & d_{12} & \dots & d_{1M} \\ d_{21} & d_{22} & \dots & \dots \\ \dots & \dots & \dots & \dots \\ d_{M1} & \dots & \dots & d_{MM} \end{matrix}$$

Una vez hemos calculado la matriz de distancia entre los centroides procedemos a calcular el peso de los clusters dentro y fuera del contorno. Para cada cluster i = 1, 2 ,…, M definimos estos pesos como:

$W_i^{INT} = \sum p_{int\_i} / \sum p\phi_{int}$; donde, $p_{int\_i} = \{p \subset (\phi_{int} \cap Cluster_i)\}$ y $p\phi_{int} = \{p \subset \phi_{int}\}$

$W_i^{EXT} = \sum p_{ext\_i} / \sum p\phi_{ext}$; donde, $p_{ext\_i} = \{p \subset (\phi_{ext} \cap Cluster_i)\}$ y $p\phi_{ext} = \{p \subset \phi_{ext}\}$

Siendo $p\phi_{int}$, los puntos interiores al contorno, $p\phi_{ext}$ los puntos exteriores y *p* los mxn puntos que conforman la imagen.

Tras la ejecución de todos estos pasos estamos en condiciones de ejecutar el algoritmo principal, la ejecución de dicho algoritmo consta de los siguientes pasos:

1. Cálculo de la sensibilidad mediante la ejecución de la función EMD-Sensibility*[1]. Con la ejecución de esta función obtenemos los parámetro de sensibilidad K_i^I y K_i^O, i = 1,…,M.

   Esta función toma como parámetros de entradas los valores de los pesos (W_i^I e W_i^O) de cada cluster y la matriz de distancias entre los centroides *[d]*.

---

*[1] Cálculo de la sensibilidad: para obtener los valores de sensibilidad usamos una implementación del algoritmo de optimización simplex (función linprogC.m)

$$\min_x \{f' \cdot x\}, \text{ con las restricciones } A \cdot x <= B$$

En este caso:

$x = [K_1^I, K_2^I, \dots, K_M^I, K_1^O, K_2^O, \dots, K_M^O]$

$f = [d_{11}, d_{12}, \dots, d_{1M}, d_{21}, d_{22}, \dots, d_{2M}, d_{31}, \dots, d_{MM}]$

A = matriz de coeficientes de orden M

$B = [W_1^I, W_2^I, \dots, W_M^I, W_1^O, W_2^O, \dots, W_M^O] = [W^I \mid W^O]$

2. Cálculo de las áreas dentro y fuera del contorno ϕ (A_I y A_O):

   $A_i = \sum p_{int} / \sum p$; donde, $p_{int} = \{p \subset \phi_{int}\}$ (puntos interiores al contorno)

   $A_o = \sum p_{ext} / \sum p$; donde, $p_{ext} = \{p \subset \phi_{ext}\}$ (puntos exteriores al contorno)

   Siendo *p* los m·n puntos que conforman la imagen, de forma que $\sum p = m \cdot n$ ($A_I + A_O = 1$).

3. Cálculo de la fuerza

$$F(\vec{x}) = -\frac{1}{A_I}\sum_{i=1}^{M} k_i^I (C_i(\vec{x}) - w_i^I) + \frac{1}{A_O}\sum_{i=1}^{M} k_i^O (C_i(\vec{x}) - w_i^O)$$

   En la expresión anterior, la función $C_i(x)$ es tal que:

   Ci(**x**) = 1 si el punto x pertenece al cluster i y 0 e.o.c.

4. Contour fast-evolution. A continuación aplicamos el algoritmo de evolución del contorno, en función de los valores del campo de fuerza F(**x**). El funcionamiento del algoritmo, en la iteración N, sería el siguiente:

   Sea ϕ_N el contorno, en la iteración N, de forma que para el punto **x** de la imagen se define como:

$$\phi_N(\mathbf{x}) = \begin{cases} 3 & \text{, si x pertenece al exterior del contorno } (\subset \Omega_{ext}) \\ 1 & \text{, si x pertenece al borde exterior del contorno } (\subset L_{ext}) \\ -1 & \text{, si x pertenece al borde interior del contorno } (\subset L_{int}) \\ -3 & \text{, si x pertenece al interior del contorno } (\subset \Omega_{int}) \end{cases}$$

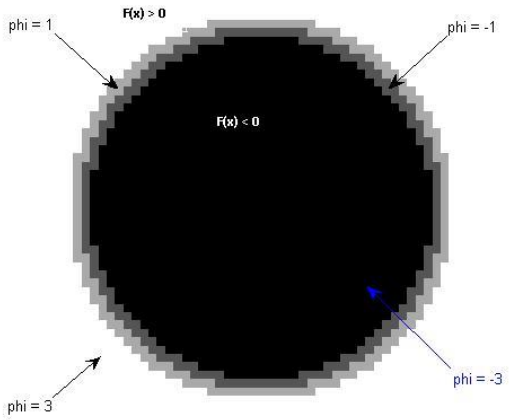


Fig. 3. Representación de $\phi_N(x)$.

Por otra parte definimos los puntos vecinos a uno dado **x** ($\subset \Re^k$, por simplicidad, supondremos k=2), como aquellos que verifican la relación:

$$N(\mathbf{x}) = \{\mathbf{y} \in D \mid \sum_k abs(y_k - x_k) = 1\}$$

Una vez definida las funciones $\phi_N(\mathbf{x})$ y $N(\mathbf{x})$ tenemos que el contorno evolucionará siguiendo las siguientes reglas:

- $\forall\, \mathbf{x} \in L_{out}$, $\mathbf{x}$: $L_{int} \rightarrow L_{out}$, si $F(\mathbf{x}) > 0$
- $\forall\, \mathbf{x} \in L_{int}$, si $\forall\, \mathbf{y} \in N(\mathbf{x})$, $\phi(\mathbf{y}) < 0$, entonces $\mathbf{x}$: $L_{int} \rightarrow \Omega_{int}$, y actualizar $\phi(\mathbf{x}) = -3$
- $\forall\, \mathbf{x} \in L_{int}$, $\mathbf{x}$: $L_{out} \rightarrow L_{int}$, si $F(\mathbf{x}) < 0$
- $\forall\, \mathbf{x} \in L_{out}$, si $\forall\, \mathbf{y} \in N(\mathbf{x})$, $\phi(\mathbf{y}) > 0$, entonces $\mathbf{x}$: $L_{out} \rightarrow \Omega_{out}$, y actualizar $\phi(\mathbf{x}) = 3$

5. Smoothing evolution. En este punto, se define el filtro gaussiano:

$$G(\mathbf{x}) = [1 / 2\pi\sigma^2] \exp(-|\mathbf{x}|^2 / 2\,\sigma^2)$$

De forma que la evolución del contorno es tal que sea:

$$\phi'_{N+1}(\mathbf{x}) = \phi_N(\mathbf{x}) \otimes G(\mathbf{x})$$

Seguimos los siguientes pasos para conformar $\phi_{N+1}(\mathbf{x})$

- $\forall\, \mathbf{x} \in L_{out}$, $\mathbf{x}$: $L_{int} \rightarrow L_{out}$, si $\phi'_{N+1}(\mathbf{x}) > 0$
- $\forall\, \mathbf{x} \in L_{int}$, si $\forall\, \mathbf{y} \in N(\mathbf{x})$, $\phi_{N+1}(\mathbf{y}) < 0$, entonces $\mathbf{x}$: $L_{int} \rightarrow \Omega_{int}$ y $\phi_{N+1}(\mathbf{x}) = -3$
- $\forall\, \mathbf{x} \in L_{int}$, $\mathbf{x}$: $L_{out} \rightarrow L_{int}$, si $\phi'_{N+1}(\mathbf{x}) < 0$
- $\forall\, \mathbf{x} \in L_{out}$, si $\forall\, \mathbf{y} \in N(\mathbf{x})$, $\phi_{N+1}(\mathbf{y}) > 0$, entonces $\mathbf{x}$: $L_{out} \rightarrow \Omega_{out}$ y $\phi_{N+1}(\mathbf{x}) = 3$

6. Cálculo de los nuevos pesos. Tras la modificación del contorno recalculamos los pesos

$$[W_1^{INT}, W_2^{INT}, \ldots, W_M^{INT}, W_1^{EXT}, W_2^{EXT}, \ldots, W_M^{EXT}]$$

7. Representación del contorno.

8. Volvemos al punto [1] si no se cumple alguna de las dos condiciones de salida del bucle principal. Estas dos condiciones son:

   a. Condición de convergencia: $\phi_N(\mathbf{x}) = \phi_{N-1}(\mathbf{x})$. Es decir, consideraremos que hemos conseguido la convergencia si el contorno no evoluciona entre una iteración y la anterior.
   b. Condición de fin por máximo número de iteraciones. En todos los casos no es posible llegar a la condición de convergencia. Típicamente se da esta situación cuando una iteración y la siguiente constituyen dos mínimos relativos de la función de sensibilidad, en esos casos el contorno oscila entre dichos mínimo, sin solución de continuidad. Por tanto es necesario imponer un número máximo de iteraciones, de forma que una vez se alcanza dicho número máximo se interrumpe la ejecución del algoritmo.

3.2 MATCH TO TEMPLATE OPERATION MODE

Este método de funcionamiento se fundamenta en los mismos principios que el anterior, aunque con ligeras variaciones. En primer lugar veremos que podemos trabajar de tres formas distintas en este modo de funcionamiento, en función del tipo de plantilla que elijamos. Estas tres formas pueden ser:

- Match to interior-points Template (M2T-I)
- Match to exterior-points Template (M2T-O)
- Match to interior/exterior-points Template (M2T-B)

Cada forma de funcionamiento se diferencia por el tipo de plantilla que definimos, estas pueden ser:

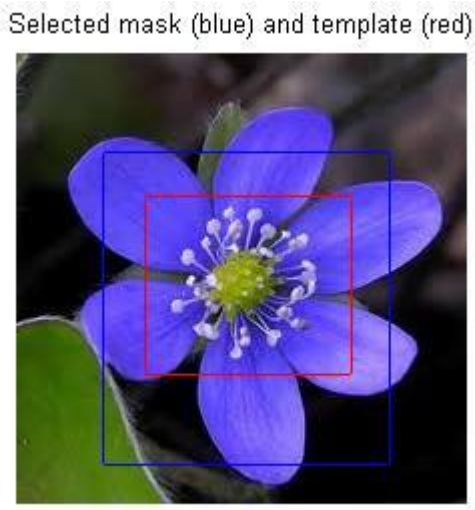


Fig. 4.A Template M2T-I

Fig. 4.B Template M2T-O

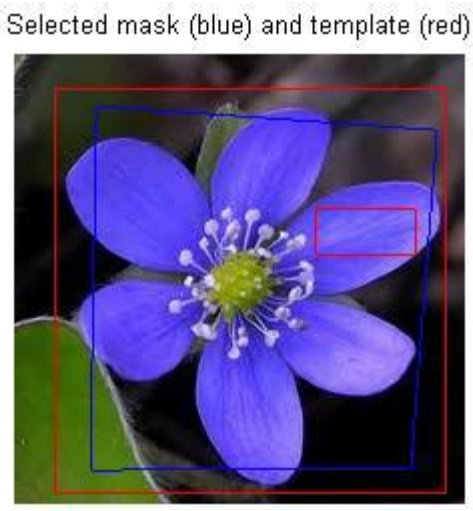


Fig. 4.C Template M2T-B

Al igual que en el caso anterior los primeros pasos también consisten en la transformación de la imagen al espacio de color L*a*b*, determinación de los clusters y sus centroides, cálculo de la distancia CIE96 entre dichos centroides (generación de la matriz de distancia) y definición del contorno inicial. Adicionalmente, antes de definir el contorno inicial se define la plantilla, la cual, como hemos visto, puede ser de tres formas distintas.

Una vez definidas la plantilla y el contorno inicial, la ejecución del algoritmo continúa con el cálculo de los pesos. Este cálculo varía en función del tipo de M2T que estemos ejecutando:

- M2T-I: (para i = 1,2, …, M)

$W_i^{INT} = \sum(p_{int})_i / \sum pT_{int}$; donde, $(p_{int})_i = \{p \subset (T_{int} \cap Cluster_i)\}$ y $pT_{int} = \{p \subset T_{int}\}$

$W_i^{EXT} = \sum(p_{ext})_i / \sum p\phi_{ext}$; donde, $(p_{ext})_i = \{p \subset (\phi_{ext} \cap Cluster_i)\}$ y $p\phi_{ext} = \{p \subset \phi_{ext}\}$

De forma que: $W = [W_1^{INT}, W_2^{INT}, \ldots, W_M^{INT}, W_1^{EXT}, W_2^{EXT}, \ldots, W_M^{EXT}] = [W^I \mid W^O]$

Siendo pT_{int}, los puntos interiores a la plantilla, $p\phi_{ext}$ los puntos exteriores al contorno y *p* los mxn puntos que conforman la imagen. La plantilla es fija y no varía con las distintas iteraciones del algoritmo, es decir, los distintos W_i^{INT} son constantes.

- M2T-O: (para i = 1,2, …, M)

$W_i^{INT} = \sum(p_{int})_i / \sum p\phi_{int}$; donde, $(p_{int})_i = \{p \subset (\phi_{int} \cap Cluster_i)\}$ y $p\phi_{int} = \{p \subset \phi_{int}\}$

$W_i^{EXT} = \sum(p_{ext})_i / \sum pT_{ext}$; donde, $(p_{ext})_i = \{p \subset (T_{ext} \cap Cluster_i)\}$ y $pT_{ext} = \{p \subset T_{ext}\}$

De forma que: $W = [W_1^{INT}, W_2^{INT}, \ldots, W_M^{INT}, W_1^{EXT}, W_2^{EXT}, \ldots, W_M^{EXT}] = [W^I \mid W^O]$

Siendo pT_{ext}, los puntos exteriores a la plantilla, $p\phi_{int}$ los puntos interiores al contorno y *p* los mxn puntos que conforman la imagen. La plantilla es fija y no varía con las distintas iteraciones del algoritmo, es decir, los distintos W_i^{EXT} son constantes.

- M2T-B: (para i = 1,2, …, M)

Fig. 5. Contorno, plantillas y zonas delimitadas

Para ilustrar este caso de forma sencilla procederemos a identificar las distintas zonas de interés usando la para ello las distintas regiones marcadas en la imagen anterior (*Fig. 5*). En dicha imagen podemos identificar las siguientes zonas:

- Zona 1 : Conformada por los puntos exteriores a la plantilla exterior (T_{ext}) estos puntos los denominaremos pT_{ext}.
- Zona 2: Conformada por los puntos interiores a la plantilla interior (T_{int}) estos puntos los denominaremos pT_{int}.
- Zona 3: Conformada por los puntos pertenecientes a la intersección de los puntos interiores a la plantilla exterior y los exteriores al contorno ($T_{EXT}^{INT} \cap \phi_{ext}$). Estos puntos los denominaremos $pT\phi^{EXT}$.
- Zona 4: Conformada por los puntos pertenecientes a la intersección de los puntos exteriores a la plantilla interior y los interiores al contorno ($T_{INT}^{EXT} \cap \phi_{int}$). Estos puntos los denominaremos $pT\phi^{INT}$.

Una vez diferenciadas las diferentes particiones presentes en la imagen procedemos a definir los pesos:

$W_i^{INT} = W_i^{Z2} + W_i^{Z4}$ siendo:

$W_i^{Z2} = \sum(pT_{int})_i / \sum pT_{int}$ donde: $(pT_{int})_i = \{pT_{int} \cap Cluster_i\}$

$W_i^{Z4} = \sum(pT\phi^{INT})_i / \sum pT\phi^{INT}$ donde: $(pT\phi^{INT})_i = \{pT\phi^{INT} \cap Cluster_i\}$

$W_i^{EXT} = W_i^{Z1} + W_i^{Z3}$ siendo:

$W_i^{Z1} = \sum pT_{ext}{}^{i} / \sum pT_{ext}$ donde: $(pT_{int})_i = \{pT_{ext} \cap Cluster_i\}$

$W_i^{Z3\_\phi} = \sum(pT\phi^{EXT})_i / \sum pT\phi^{EXT}$ donde: $(pT\phi^{EXT})_i = \{ pT\phi^{EXT} \cap Cluster_i\}$

Debido a que las plantillas T_{EXT} y T_{INT} no varían durante la ejecución del algoritmo tenemos que las componentes W_i^{Z1} y W_i^{Z2} de los pesos son constantes.

Finalmente, el vector de pesos se conformará como sigue:

$$W = [W_1^{INT},\ldots, W_M^{INT}, W_1^{EXT},\ldots, W_M^{EXT}] = [W^I \mid W^O]$$

Una vez hemos definido los pesos pasamos a la ejecución del algoritmo principal, la cual también variará en función del tipo de plantilla que hallamos escogido. Con respecto al modo de ejecución Maximal Discrepancy, la única diferencia consiste en la definición de la fuerza y los pesos. Como las diferencias en el cálculo de los pesos ya las hemos definido en el punto anterior, nos limitaremos aquí a mostrar las diferentes definiciones del campo de fuerza en función del tipo de plantilla elegido.

- M2T-I: $$F(\vec{x}) = \frac{1}{A_O}\sum_{i=1}^{M} k_i^O (C_i(\vec{x}) - w_i^O)$$

- M2T-O: $$F(\vec{x}) = -\frac{1}{A_I}\sum_{i=1}^{M} k_i^I (C_i(\vec{x}) - w_i^I)$$

- M2T-B: $$F(\vec{x}) = -\frac{1}{A_I}\sum_{i=1}^{M} k_i^I (C_i(\vec{x}) - w_i^I) + \frac{1}{A_O}\sum_{i=1}^{M} k_i^O (C_i(\vec{x}) - w_i^O)$$

Las distintas configuraciones de las fuerzas en función de modo M2T elegido se deben a que el en los casos en los que el vector de peso es constante el resultado de los valores de sensibilidad asociados es nulo. Esto es así debido a que los $K_i^{I/O}$ no son más que la aproximación (linearizada) del gradiente de la EMD dentro y fuera del contorno con respecto a la variación de este:

$$\frac{\partial EMD}{\partial \phi} = \frac{\partial EMD}{\partial W} \bullet \frac{\partial W}{\partial \phi}$$

3.3 ALGORITMO DE AUTOINICIALIZACIÓN

En este apartado describiremos el funcionamiento del algoritmo de autoinicialización, este algoritmo obtiene de forma automática un contorno inicial sobre el cual empieza a ejecutarse el algoritmo de segmentación.

Supongamos que nuestra imagen tiene dimensiones W x H, por simplicidad supondremos que W>H (en caso contrario una transposición espacial, de 90º, de la imagen conseguiría que se cumpliese esta relación). Entonces, sea M la densidad de clusters deseada hacemos un partición de la imagen en M x M clusters rectangulares (o cuadrados si W = H) de dimensiones W/M y H/M. Para cada uno de estos clusters definiremos su centro, QC_i, en el punto medio de los mismos.

Si asumimos que previamente hemos aplicado el algoritmo kmeans (con N clusters) en la transformación L*a*b* de la imagen (descrito en el punto 2.1), estamos en condiciones de definir para cada cluster Ci (i = 1, …, MxM) el vector Q_i, de longitud N en el cual el elemento j de dicho vector contendrá el número de pixeles del k-$cluster_j$ (j = 1, …, N) contenidos en el cluster C_i.

Una vez definidos los $\{Q_1, Q_2, \ldots, Q_{MxM}\}$ el algoritmo, el cual seguirá la siguiente secuencia de pasos:

(1). Seleccionamos de forma aleatoria dos centros C_1 y C_2 del conjunto $\{Q_1, \ldots, Q_{MxM}\}$

(2). Para cada Q_i ($i \neq 1,2$) evaluamos y comparamos las expresiones $EMD(Q_i, C_1)$ y $EMD(Q_i, C_2)$. Asignando el cluster Qi a la partición representada por C_1 o C_2 siguiendo la siguiente regla:

$\forall$ i ($i \neq 1,2$) si $EMD(Q_i, C_1) > EMD(Q_i, C_2)$ asignar Q_i a C_2
$\forall$ i ($i \neq 1,2$) si $EMD(Q_i, C_1) < EMD(Q_i, C_2)$ asignar Q_i a C_1

(3). Recalculamos los valores de C1 y C2 tras las asignaciones

(4). Volvemos al punto (2) hasta conseguir la estabilización del algoritmo (C_1 y C_2 no varían entre una iteración y la siguiente.

(5). Obtener el valor de $DE_r = abs[EMD(C_1, C_2)]$ (r = 1, 2, …,R)

Ejecutaremos este secuencia de pasos R veces, obteniendo en consecuencia R particiones. Finalmente la tomaremos como contorno inicial la partición r, tal que $argmax_{(r)}(DE_r)$.

En las siguiente imágenes mostramos el resultado de la aplicación del algoritmo para diferentes valores M con R = 5 en todos los casos.

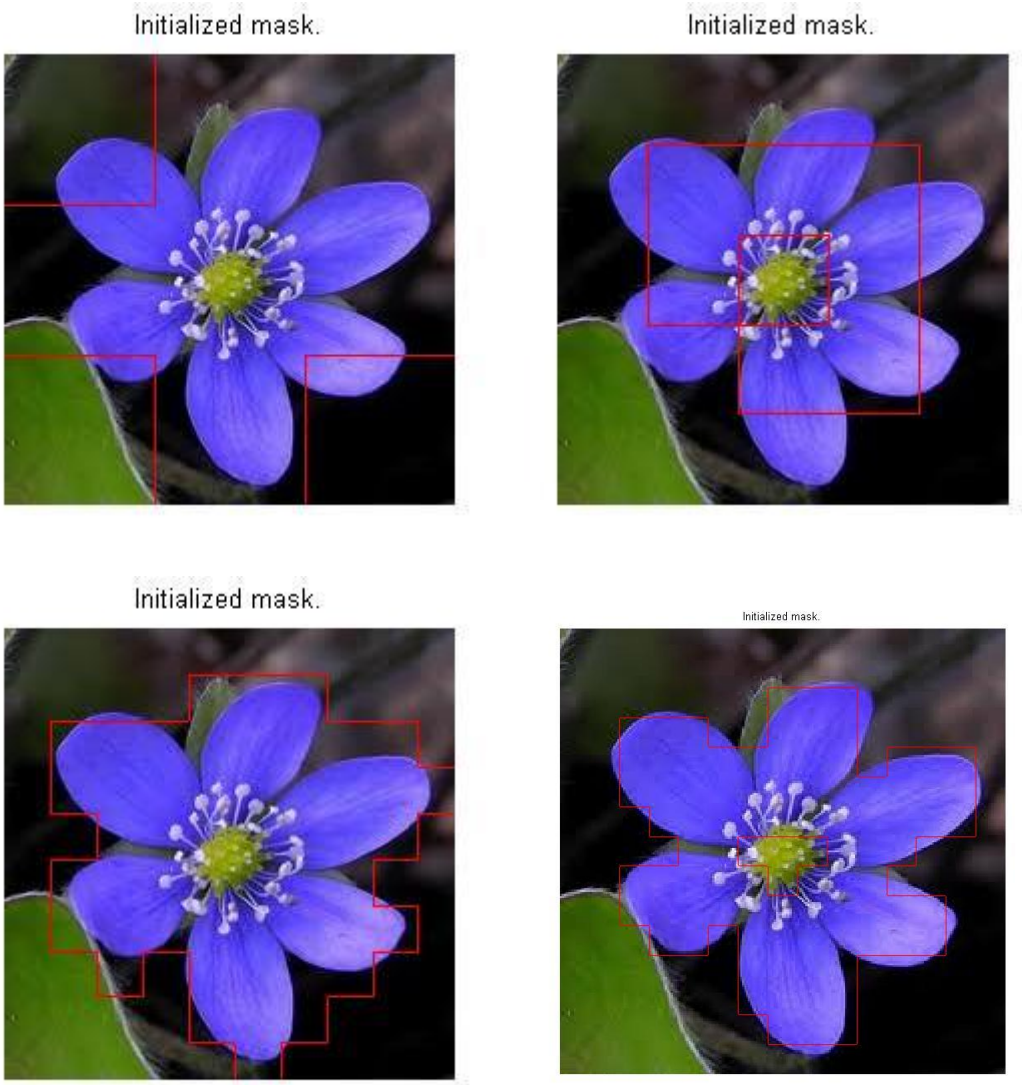
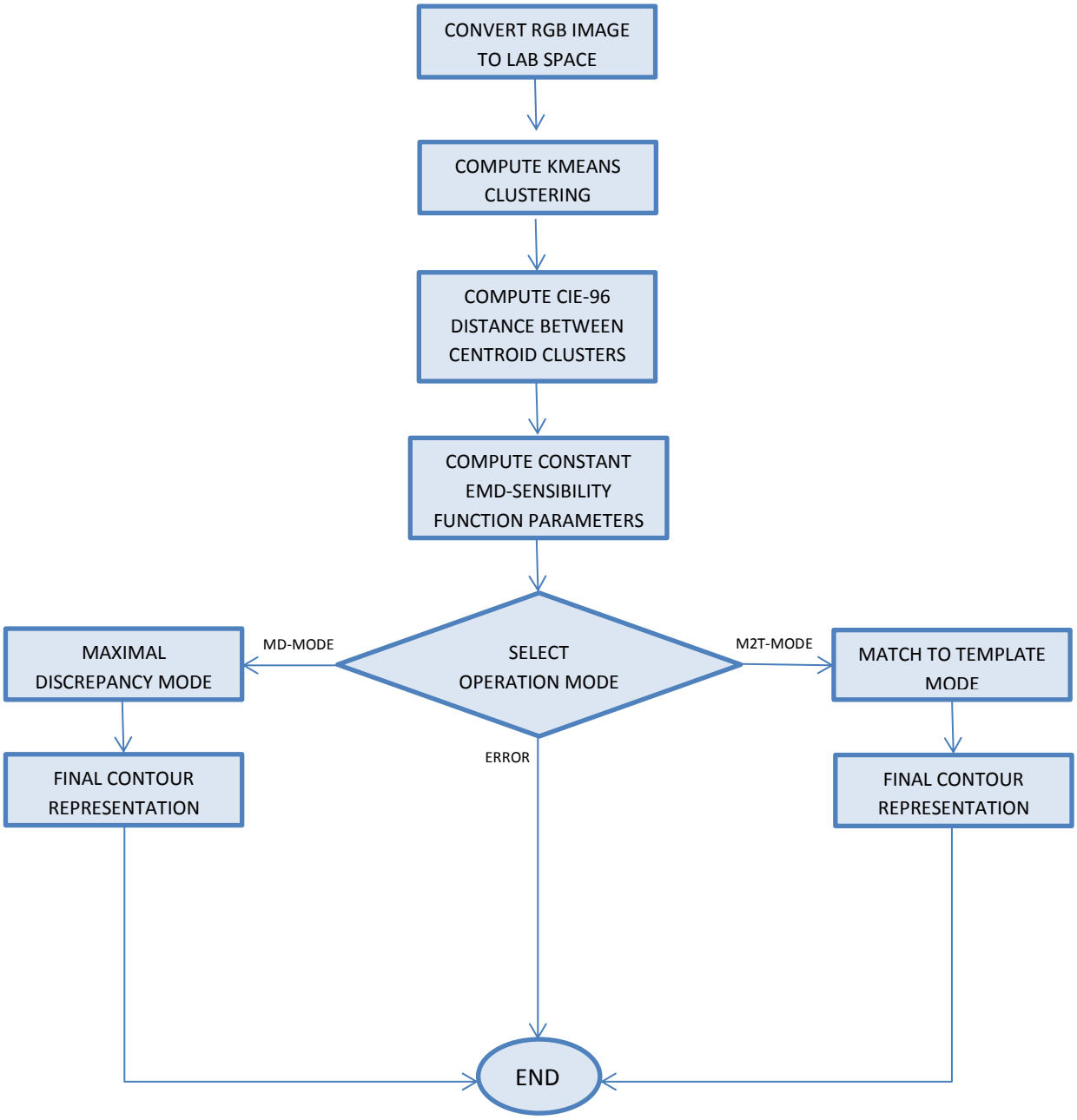


Fig. 6. Resultado algoritmo de inicialización con M = 3, 5, 10 y 15

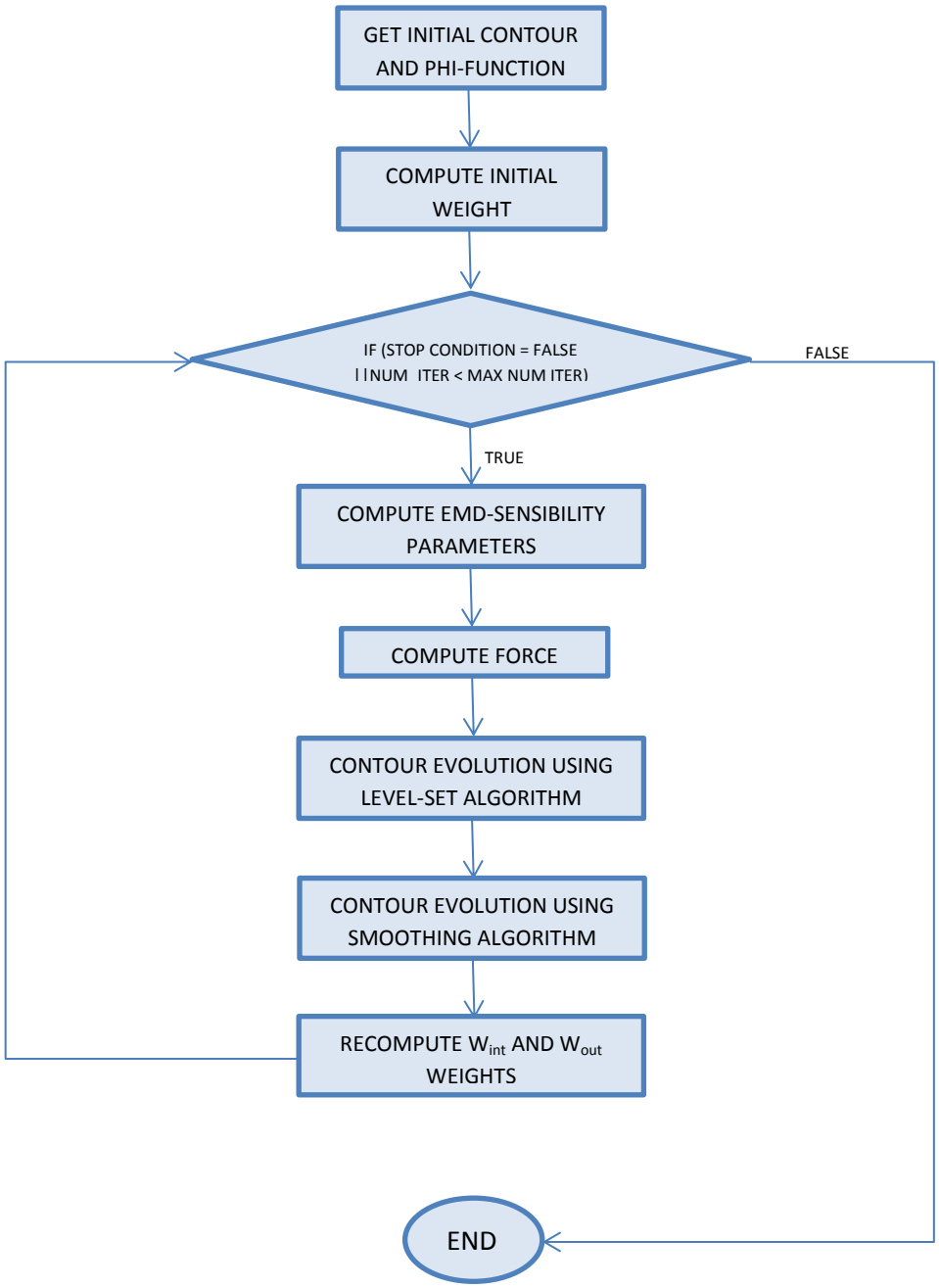
4 IMPLEMENTACIÓN DEL ALGORITMO (DIAGRAMAS DE FLUJO)

En esta sección daremos una visión esquemática del funcionamiento del algoritmo, para ello, usamos principalmente diagramas de flujo. La lista completa de funciones y las relaciones de dependencia (parent – children) definidas entre ellas se encuentran definidas, de forma detallada en al *ANEXO 2 (DEPENDENCY REPORT)*, de la misma forma el código *MATLAB* sobre el que se construye el algoritmo está recogido en el *ANEXO 1 (CÓDIGOS MATLAB).*

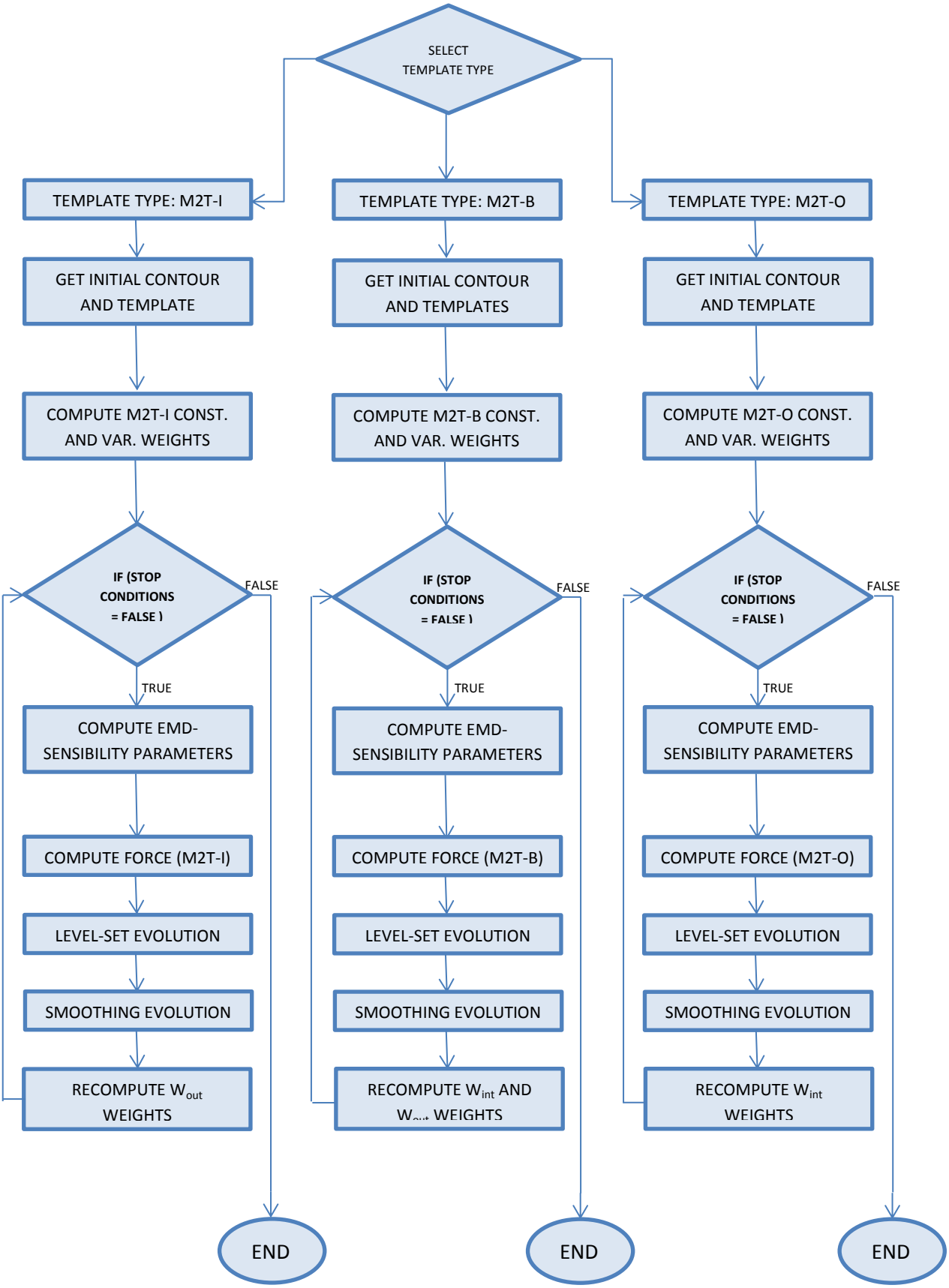
4.1 FUNCIÓN PRINCIPAL (ESQUEMA GENERAL):



4.2 FUNCIÓN PRINCIPAL (MAXIMAL-DISCREPANCY OPERATION MODE):



4.3 FUNCIÓN PRINCIPAL (MATCH-TO-TEMPLATE OPERATION MODE):



5 RESULTADOS EXPERIMENTALES

Sean D, por simplicidad, perteneciente a R^2, de tamaño m x n, el soporte sobre el cual se define la imagen a segmentar y Ω la región objetivo (perteneciente a D). Sobre dicho soporte, D, podemos trazar las siguientes curvas:

T ≡ {Puntos que definen la plantilla}

C ≡ {Puntos que definen el contorno}

Estas curvas permiten subdividir, de forma independiente, el soporte D, en dos regiones disjuntas,

C_{int} ≡ {Región interior al contorno C}

C_{ext} ≡ {Región exterior al contorno C}

T_{int} ≡ {Región interior a la plantilla T}

T_{ext} ≡ {Región exterior a la plantilla T}

De forma que $T_{int} \cup T_{ext} = D$ y $C_{int} \cup C_{ext} = D$. Por otra parte definida EMD(X, Y) como la función que evalúa la distancia, en términos de Perceptual Earth Mover's Distance, entre las regiones X e Y, tenemos que en función de los diferentes modos de funcionamiento del algoritmo podemos discernir los siguientes casos:

- Caso 1: Match to interior-points template (M2T-I). En esta situación, sean T y C, las curvas, definidas por el usuario, que definen la plantilla y contorno iniciales. Tenemos que la plantilla se corresponderá con los puntos pertenecientes a la región T_{int}. En este escenario la curva T ha de estar inscrita dentro de la región objetivo (Ω), es decir, $T_{int} \subset \Omega$. Por otro lado, C y T han de ser tal que $T_{int} \cap \Omega = \Phi$ y $C_{int} \cap \Omega = \Phi$ y $C_{int} \cap T_{int} = \Phi$. En estas condiciones el algoritmo evoluciona en cada iteración de forma que se maximice la relación:

$$\boldsymbol{max\ \{EMD\ (T_{int}) - EMD\ (C_{int})\} = max\ \{EMD\ (T_{int},\ C_{int})\}}$$

- Caso 2: Match to exterior-points template (M2T-O). En esta situación, sean T y C, las curvas, definidas por el usuario, que definen la plantilla y contorno iniciales. Tenemos que la plantilla se corresponderá con los puntos pertenecientes a la región T_{ext}. En este escenario la curva T ha de estar circunscrita a la región objetivo (Ω), es decir, $\Omega \subset T_{int}$. Por otro lado, C y T han de ser tal que $T_{ext} \cap \Omega = \Phi$ y $C_{ext} \cap \Omega = \Phi$ y $C_{int} \cap T_{int} \neq \Phi$ ($C_{int} \subset T_{int}$). En estas condiciones el algoritmo evoluciona en cada iteración de forma que se maximice la relación:

$$\boldsymbol{max\ \{EMD\ (T_{ext}) - EMD\ (C_{int})\} = max\ \{EMD\ (T_{ext},\ C_{int})\}}$$

- Caso 3: Match to interior/exterior-points template (M2T-B). En este caso se definen dos plantilla T_1 y T_2, tales que $T_{1ext} \cap T_{2int} = \Phi$, $\Omega \subset T_{1int}$ y $T_{2int} \subset \Omega$. Por otra

parte el contorno inicial ha de cumplir que $C \subset T_{1int} \cap T_{2ext}$. En estas condiciones la ejecución del algoritmo nos lleva a maximizar la expresión:

$$\boldsymbol{max \{EMD (T_{1int} \cup T_{2ext}) - EMD (C_{int} \setminus T_{1int})\} = max \{EMD ((T_{1int} \cup T_{2ext}), (C_{int} \setminus T_{1int})\}}$$

- Caso 4: Auto-initialized Maximal Discrepancy (AI-MD). El algoritmo de autoinicialización proporciona de manera automática un contorno inicial, C. Una vez definido este contorno inicial la ejecución del algoritmo permite maximizar la relación:

$$\boldsymbol{max \{|EMD (Cint) - EMD (Cext)|\} = max \{EMD (Cint, Cext)\}}$$

- Caso 5: User-initialized Maximal Discrepancy (UI-MD). En este modo de funcionamiento, se traza un contorno inicial, C. El algoritmo tiende a la convergencia maximizando la relación:

$$\boldsymbol{max \{|EMD (Cint) - EMD (Cext)|\} = max \{EMD (Cint, Cext)\}}$$

Una vez definido los diferentes modos de funcionamiento del algoritmo procedemos a presentar los diferentes resultados gráficos, asociados cada uno de estos modos de funcionamiento anteriormente descritos.

5.1 MATCH TO TEMPLATE (MATCH TO INTERIOR-POINTS TEMPLATE, M2T-I)

Este primer caso muestra la ejecución del algoritmo usando el modo de funcionamiento M2T-I. El recuadro rojo muestra la máscara, que delimita la plantilla (template) que estará definida por los puntos interiores a dicho contorno. El recuadro azul define el contorno inicial sobre el cual empezará a ejecutarse el algoritmo.

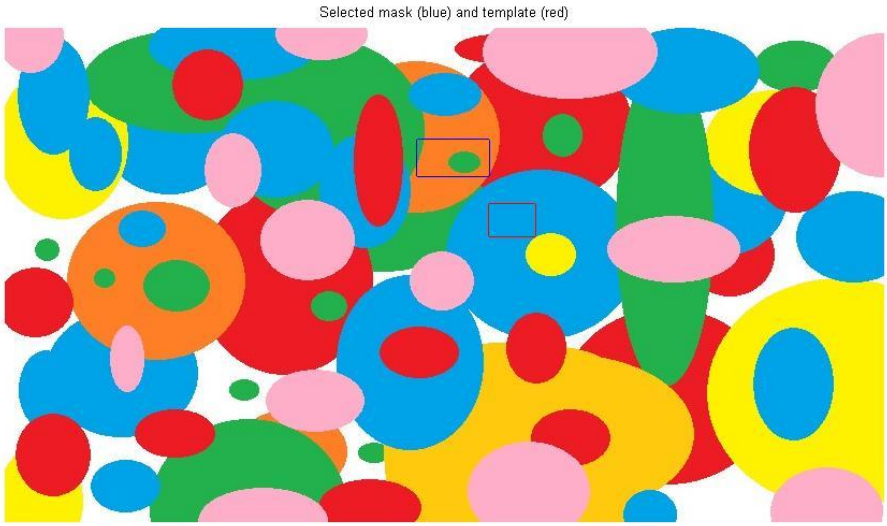


Fig. 7. Initial contour (blue) and M2T-I template (red).

El resultado, tras 437 iteraciones del algoritmo, es el que se muestra en la siguiente imagen.

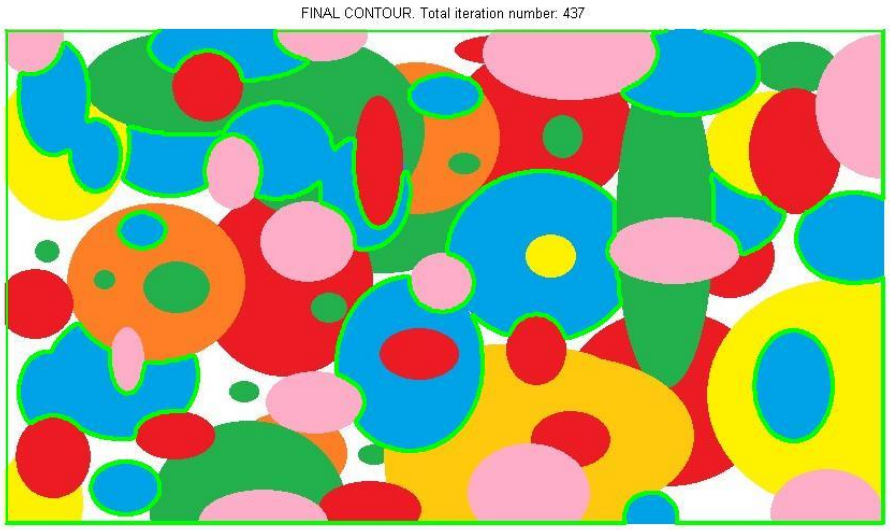


Fig. 8. Final contour (green). Segmented image.

Para ilustrar el funcionamiento del algoritmo, usando el modo de funcionamiento M2T-I, y la dependencia del resultado final, en función de la plantilla inicial trazada; a continuación mostramos el resultado de la ejecución del algoritmo sobre la misma imagen con otras condiciones iniciales.

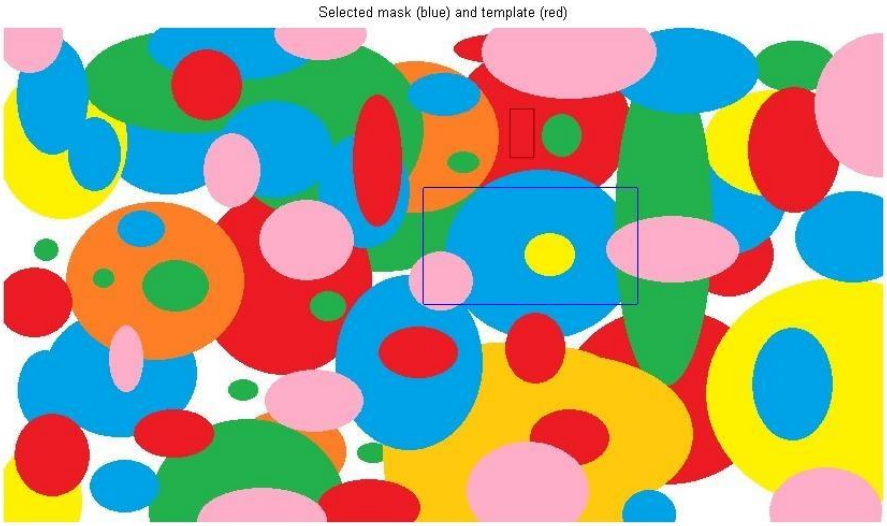


Fig. 9. Initial contour (blue) and M2T-I template (red).

El resultado final, con esta inicialización es el siguiente:

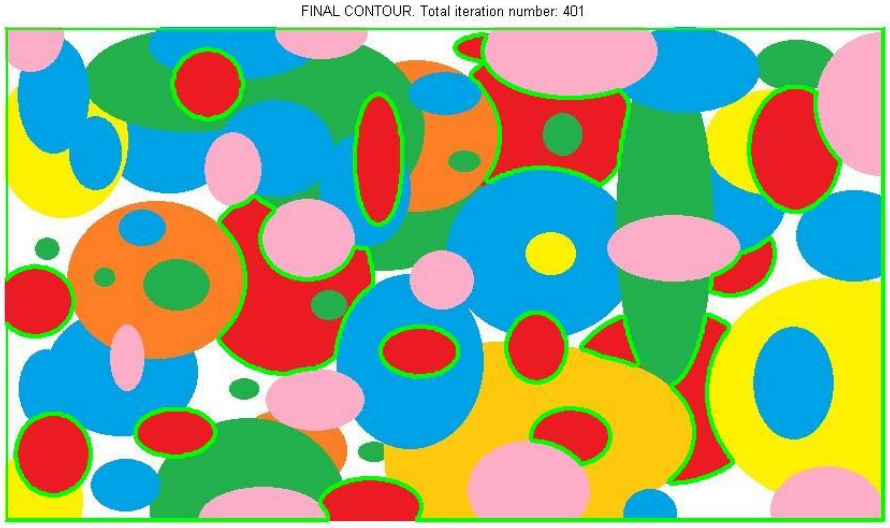


Fig. 10. Final contour (green). Segmented image.

En este segundo ejemplo, en comparación con el anterior, se observa de forma precisa como el contorno final depende de la plantilla inicial trazada. De forma que la región de la imagen exterior al contorno es aquella que presenta una máxima discrepancia, en términos de perceptual EMD con respecto a la región contenida en el interior de la plantilla.

5.2 MATCH TO TEMPLATE (MATCH TO EXTERIOR-POINTS TEMPLATE, M2T-O)

A continuación mostramos el resultado de la ejecución del algoritmo usando el modo de funcionamiento M2T-O. En este caso, la plantilla estará definida por los puntos exteriores al recuadro rojo y el contorno inicial vendrá dado por el recuadro azul.

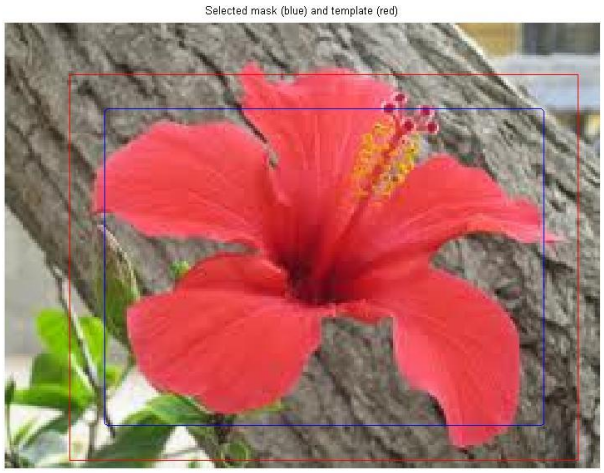


Fig. 11. Initial contour (blue) and M2T-O template (blue).

El resultado tras la ejecución del algoritmo es el siguiente:

Fig. 12. Final contour. Segmented image.

El contorno final separa la imagen dos regiones de forma, la región interior al contorno presenta la máxima discrepancia, en términos de Perceptual Earth Mover's Distance, con respecto a la región exterior al recuadro que define la plantilla.

5.3 MATCH TO TEMPLATE (MATCH TO INTERIOR/EXTERIOR-POINTS TEMPLATE, M2T-B)

Para ilustrar este método de funcionamiento, presentamos las siguientes imágenes. El método MT2-B funciona de forma que el contorno es aquel que presenta una máxima discrepancia tanto con el exterior de la plantilla 1 (cuadro negro exterior) como con el interior de la plantilla 2 (cuadro negro interior). En este caso el contorno inicial viene dado por el recuadro azul.

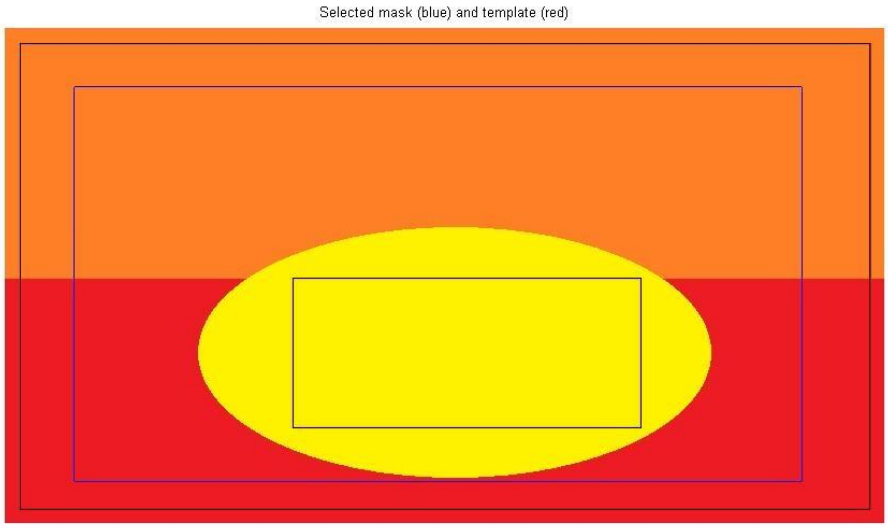


Fig. 13. Initial contour (blue) and M2T-B templates (black).

Tras la ejecución del algoritmo el contorno final es el siguiente:

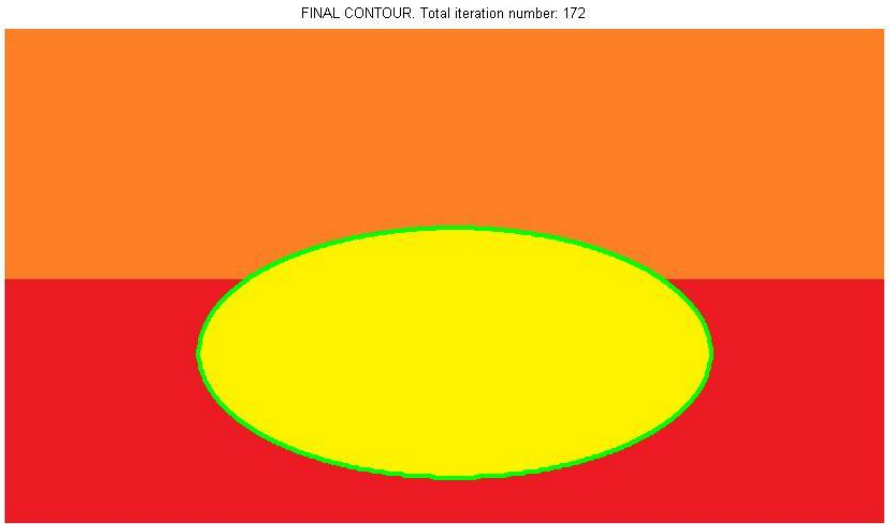


Fig. 14. Final contour (green). Segmented image.

Para ver cómo funciona realmente este método compararemos el resultado de la segmentación, partiendo del mismo contorno inicial, con el modo de funcionamiento Maximal Discrepancy.

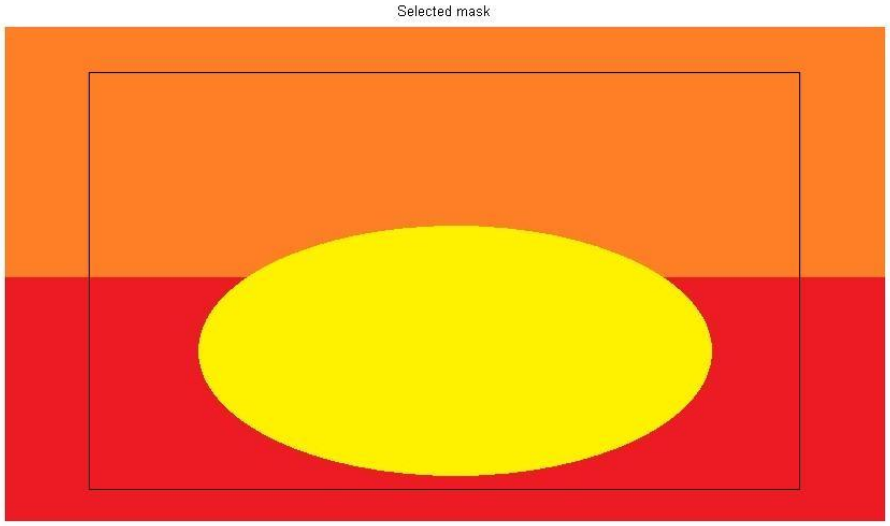


Fig. 15. Initial contour (blue).

El resultado de la segmentación, en este modo de funcionamiento, es el siguiente:

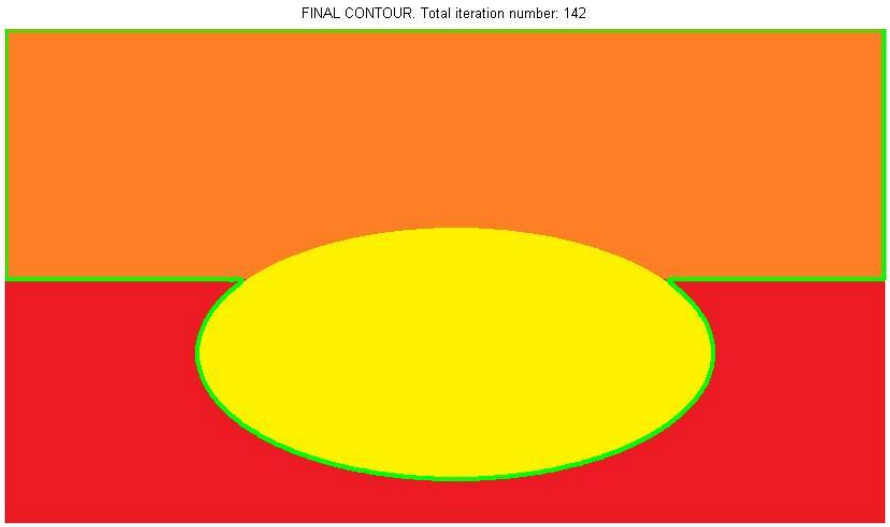


Fig. 16. Final contour (green). Segmented image.

Se aprecia como, aún partiendo de las mismas condiciones iniciales (mismo contorno inicial), el resultado es diferente si ejecutamos el modo de funcionamiento Maximal Discrepancy. Esto ocurre porque en este modo no imponemos ninguna restricción adicional a la evolución del contorno.

5.4 AUTO-INITIALIZED MAXIMAL DISCREPANCY (AI-MD):

A continuación, con el siguiente ejemplo, mostramos el funcionamiento del algoritmo y de la autoinicialización del contorno. La siguiente imagen nos permite ver el resultado del algoritmo de autoinicialización, el cual en este ejemplo, genera el contorno inicial que puede observarse marcado sobre la imagen (en rojo):

Fig. 17. Auto-initialized contour (red).

El resultado final, tras la ejecución del algoritmo es el siguiente:

Fig. 18. Final contour (green). Segmented image.

Este método, en la mayoría de los casos, suele obtener la convergencia del algoritmo con mayor rapidez, ya que inicialmente delimitamos la máscara que presenta mayor discrepancia (en términos de perceptual earth mover's distance) entre los puntos interiores a la misma y los exteriores.

5.5 USER-INITIALIZED MAXIMAL DISCREPANCY (UI-MD)

Este último ejemplo nos permite ilustrar el modo de funcionamiento más sencillo, aquel en el cual el usuario define de forma manual la máscara inicial. La imagen muestra (en rojo) el contorno inicial trazado por el usuario:

Fig. 19. Initial contour (red).

El resultado de la segmentación es, en este caso:

Fig. 20. Final contour (green). Segmented image.

Con estos ejemplos hemos mostrado los cincos modos de funcionamiento básicos del algoritmo.

6. CONCLUSIONES

A modo de conclusión hemos de destacar diferentes aspectos del trabajo anteriormente descrito. En comparación con otros métodos puede comprobarse que este algoritmo de segmentación presenta una velocidad de convergencia superior, circunstancia que permite obtener la imagen segmentada de forma más rápida reduciendo el tiempo de computación. Es interesante destacar también que la opción autoinicialización permite automatizar el trabajo de segmentado de forma que es posible programar el análisis de un conjunto de imágenes de manera que estas puedan ser procesadas de forma autónoma sin intervención directa; esta autoinicialización también facilitaría integrar el algoritmo en diferentes aplicaciones o dispositivos que hagan uso de la segmentación de imagen de forma sencilla y transparente para el usuario que haga uso de las mismas.

Por otra parte es necesario valorar la posibilidad de realizar diferentes mejoras sobre el algoritmo propuesto. Estas mejoras se pueden dividir en dos grandes grupos: computacionales y paramétricas. Dentro del grupo que hemos denominado computacionales, englobaríamos todas aquellas técnicas que permitirían mejorar la velocidad de ejecución del algoritmo; por un lado sería deseable implementar el código sobre un lenguaje de programación más eficiente que el usado en este caso. Un lenguaje de más bajo nivel, por ejemplo C+, nos permitiría con total probabilidad incrementar el rendimiento del algoritmo disminuyendo el tiempo necesario para obtener el resultado final. Además podrían emplearse técnicas de optimización del código que permitieran también disminuir el tiempo de ejecución. Respecto a las mejoras de tipo paramétrico se propone realizar, de forma empírica, un conjunto de pruebas que permitan un ajuste fino de los parámetros que intervienen en el algoritmo; evaluando la sensibilidad del modelo ante la variación de los diferentes parámetros conseguiríamos obtener unos resultados mejores, definiendo los conjuntos paramétricos estándar para diferentes grupos de imágenes homogéneas entre si sería posible generar librerías de parámetros para aplicaciones concretas.

Otra posible línea de trabajo consistiría en probar el funcionamiento del método integrando un esquema de *clustering* diferente del aquí utilizado (k-means). Esto quizás permitiría encontrar distribuciones de clusters más ajustadas al propósito que nos ocupa.

Como conclusión final, destacaría que al ser este un algoritmo de tipo generalista, pensado para procesar cualquier tipo de imagen en color y a pesar de las excelentes prestaciones del mismo es posible realizar diferentes ajustes sobre el modelo original que permitan una mejor adaptación del mismo para diferentes aplicaciones, esto junto con las posibles mejoras computacionales y de ajuste paramétrico constituyen las posibles líneas de trabajo futuro.

7. REFERENCIAS

7.1 BIBLIOGRAFÍA

[1] Yonggang Shi and William Clem Karl, "A Real-Time Algorithm for the Approximation of Level-Set-Based Curve Evolution", in *IEEE Transactions on Image Processing*, Vol. 17, No. 5, MAY 2008.

[2] Amit Adam, Ron Kimmel and Ehud Rivlin, "On Scene Segmentation and Histograms-Based Curve Evolution", in *IEEE Transactions on Patterns Analysis and Machine Intelligence*, Vol. 31, No 9, SEPTEMBER 2009.

[3] James Malcom, Yogesh Rathi, Anthony Yezzi and Allen Tannenbaum, "Fast Approximate Surface Evolution in Arbitrary Dimension", in *Proc. of SPIE*, Vol. 6914 69144C-1.

[4] Martin Maska and Pavel Matula, "A Fast Level Set-Like Algorithm with Topology Preserving Constraint", in *Springer-Verlag Berlin Heidelberg,* 2009.

[5] Francesc Serratosa and Alberto Sanfeliu, "Signature versus Histograms: Definitions, distances and algorithms", in *Pattern Recognition*, Vol. 39, pp. 921-934, 2006.

[6] Yossi Rubner, Carlo Tomasi and Leonidas J. Guibas, "The Earth Mover's Distance as a Metric for Image Retrieval", in *International Journal of Computer Vision*, Vol. 40, pp. 99-121, 2000.

[7] Kangyu Ni, Xavier Bresson and Tony Chan, "Local Histogram Based Segmentation Using tha Wasserstein Distance", in *International Journal of Computer Vision*, Vol. 84, pp. 97-111, 2009.

[8] Yossi Rubner, Jan Puzicha, Carlo Tomasi and Joachim M. Buhmann, "Empirical Evaluation of Dissimilarity Measures for Color and Texture", in *Computer Vision and Image Understanding*, Vol. 84, pp. 25-43, 2001.

[9] Qi Zhao, Zhi Yang and Hai Tao, " Differential Earth Mover's Distance with Its Applications to Visual Tracking", in *IEEE Transactions on Patterns Analysis and Machine Intelligence*, Vol. 32, No 2, FEBRUARY 2010.

[10] Oleg Michailovich, Yogesh Rathi and Allen Tannenbaum, "Image Segmentation Using Active Contours Driven by the Bhattacharyya Gradient Flow", in *IEEE Transactions on Image Processing*, Vol. 16, No. 11, NOVEMBER 2007.

[11] Tony Chan and Luminita Vese, "Active Contours without Edges", in *IEEE Transactions on Image Processing*, Vol. 10, No. 2, FEBRUARY 2002.

[12] Carlos S. Mendoza, Aurora Sáez, Begoña Acha and Carmen Serrano, "Linearized Multidimensional Earth Mover's Distance Gradient Flows", Department of Signal Processing and Communications, University of Seville, 2011.