Morfismos, Vol. 16, No. 1, 2012, pp. 9–27

Sistemas de funciones iteradas por partes*

Sergio Mabel Juárez Vázquez [1]
Flor de María Correa Romero [2]

Resumen

En la actualidad los medios digitales se han convertido en herramientas indispensables ya que la mayoría de la información se maneja mediante ellos. Las imágenes son una forma imprescindible de información pero lamentablemente ocupan bastante espacio en la memoria de una computadora y a su vez los dispositivos de almacenamiento digital suelen ser caros. Reducir el espacio que las imágenes ocupan en la memoria de una computadora baja los costos y permite que la transmisión de éstas sea más eficiente.
El objetivo de este trabajo es presentar una aplicación del concepto de conjunto fractal a la compresión de imágenes digitales. La técnica que se usará para la compresión está basada en la teoría matemática denominada sistemas de funciones iteradas por partes.

2010 Mathematics Subject Classification: 68U10, 65D18.
Keywords and phrases: sistemas de funciones iteradas, sistemas de funciones iteradas por partes, conjunto fractal, compresión fractal de imágenes.

1 Introducción

La información que actualmente se maneja es en su mayoría a través de computadoras, la importancia que ésta tiene para que los sistemas social y económico funcionen, trajo consigo que se esté constantemente desarrollando herramientas para tratarla. Problemas como el asegurar la información y almacenarla de manera digital han tomado relevancia.

*Este trabajo es parte de la tesis de licenciatura desarrollada por el primer autor bajo la dirección de la segunda autora. La tesis fue presentada en el Departamento de Matemáticas de la Escuela Superior de Física y Matemáticas del Instituto Politécnico Nacional en noviembre de 2010.

[1]Becario Conacyt.

[2]Profesora Becaria COFAA, Departamento de Matemáticas ESFM-IPN.

En el caso del almacenamiento digital de información se han creado métodos para reducir su volumen sin que se afecte el contenido de ésta. Las imágenes digitales son una gran fuente de información. Quién no ha escuchado la frase: una imagen dice más que mil palabras. Como las imágenes suelen ocupar mucho espacio en la memoria de una computadora, se han desarrollado técnicas de compresión para ayudar a resolver tal problema.

La compresión de imágenes puede ser con pérdida o sin pérdida de información. Los algoritmos que realizan la compresión con pérdida, reducen la calidad de la imagen reconstruida después del proceso de compresión, así pues esta imagen puede discrepar en relación a la imagen original en detalles como contornos, formas y colores. Con los algoritmos sin pérdida, las imágenes procesadas quedan intactas, pero pagan esta ventaja con una razón de compresión menor en comparación a los algoritmos con pérdida, por lo tanto se ocupar más memoria para almacenarlas. Para algunos casos se requiere de comprimir imágenes en donde la pérdida sea mínima o nula. Ejemplos de esto pueden ser imágenes médicas, imágenes de huellas dactilares, imágenes donde haya texto, en general imágenes cuya información contenida se necesita de forma muy precisa. En otros casos se puede aprovechar de algunas limitaciones que el ojo humano tiene, de la habilidad que hemos desarrollado para poder intuir formas, del simbolismo que asociamos a ciertas imágenes o simplemente que algunos detalles de la imagen no tienen importancia para nosotros. Si un ser querido nos muestra una foto en donde él aparece y en el fondo hay algunas nubes, uno pondrá atención primordialmente en la persona sin darle tanta importancia a la forma que cada nube tiene, además las nubes tienen formas aleatorias, así que si la imagen es procesada mediante una técnica de compresión y en el proceso se pierde información sobre la forma exacta que la nube tiene en la imagen, antes y depués del proceso, no lo notaremos o simplemente no importará, pues nos interesará sólo el hecho de que es una nube. La compresión fractal es una técnica de las que se denomina con pérdida, la imagen una vez decodificada no es igual que la imagen original.

Este trabajo ilustra la aplicación que la geometría fractal tiene a la compresión de imágenes digitales. Cada vez que se ocupe la palabra imagen se hará referencia a una imagen digital a menos que se mencione lo contrario. A lo largo de las secciones siguientes se dará un método para codificar y comprimir imágenes en escala de grises con ayuda del concepto de conjunto fractal como el atractor de un sistema de funciones iteradas por partes (SFIP). Los autores observamos que en toda la literatura que revisamos sobre compresión fractal de imágenes, el concepto de SFIP carece de una definición formal y no se comprueba que se puede inducir una contracción a través del sistema. En este artículo se propone una definición para los SFIP

y se demuestra que dado un SFIP lo podemos asociar con una contracción.

Cabe mencionar que la generalización de codificación de imágenes a colores no es difícil a partir de la teoría de codificación de imágenes en escala de grises.

2 Teoría básica

Los resultados que se mencionan en esta sección son conocidos, por lo que sólo se enunciarán. El lector puede consultar las demostraciones en [1] y [13].

Sea (X, d) un espacio métrico, denotaremos por $\mathscr{H}(X)$ al conjunto de todos los subconjuntos compactos y no vacíos de (X, d).

Definición 2.0.1. Sea (X, d) un espacio métrico, sean $x \in X$ y $A, B \in \mathscr{H}(X)$. Se define la distancia de x al conjunto A como

$$d(x, A) := min\{d(x, y) \mid y \in A\},$$

y la distancia de A a B como

$$d(A, B) := max\{d(x, B) \mid x \in A\}.$$

Proposición 2.0.2. *Sea (X, d) un espacio métrico, y sea*

$$h : \mathscr{H}(X) \times \mathscr{H}(X) \longrightarrow \mathbb{R}$$

la función definida por $\forall\ A, B \in \mathscr{H}(X)$, $h(A, B) := d(A, B) \vee d(B, A)$, entonces tenemos que h es una métrica sobre $\mathscr{H}(X)$.

Teorema 2.0.3. *Sea (X, d) un espacio métrico completo, entonces el espacio métrico $(\mathscr{H}(X), h)$ es completo.*

Definición 2.0.4. Sean (X_1, d_1) y (X_2, d_2) dos espacios métricos. Una función $f : (X_1, d_1) \longrightarrow (X_2, d_2)$ es una función de Lipschitz, si existe un número real positivo α tal que

$$\forall\ x, y \in X, \qquad d_2(f(x), f(y)) \leq \alpha\, d_1(x, y),$$

al número α se le llama un factor de Lipschitz de la función f. Si se cumple que $0 \leq \alpha < 1$, entonces f es llamada una contracción y α un factor de contracción para f.

Teorema 2.0.5. ***(Teorema del punto fijo).*** *Si (X, d) es un espacio métrico completo y $f : X \longrightarrow X$ una contracción, con α un factor de contracción de f, entonces existe un único $x_f \in X$ tal que $f(x_f) = x_f$, a x_f se le llama el punto fijo de la contracción.*

2.1 Sistemas de funciones iteradas

Proposición 2.1.1. *Sea (X,d) un espacio métrico completo y sean para $\imath \in \{1,\ldots,n\}$ $f_\imath : (X,d) \longrightarrow (X,d)$ contracciones, con $\alpha_\imath$ un factor de contracción para $f_\imath$. Sea $F : (\mathscr{H}(X),h) \longrightarrow (\mathscr{H}(X),h)$ la función definida por*

$$\forall\ A \in \mathscr{H}(X), \qquad F(A) = \bigcup_{\imath=1}^{n} f_\imath(A),$$

entonces F es una contracción y $\alpha := máx\,\{\alpha_\imath | \imath \in \{1,\ldots,n\}\}$ es un factor de contracción para F.

Definición 2.1.2. Un sistema de funciones iteradas o SFI, consiste de un espacio métrico completo (X,d) y una familia finita de contracciones

$$\{f_\imath : (X,d) \longrightarrow (X,d) \mid \imath \in \{1,\ldots,k\}\},$$

al SFI se le denota por $\{(X,d)\ ;\ f_1, f_2, \ldots, f_k\}$, y se llama un factor de contracción del SFI al número $\alpha := máx\,\{\alpha_\imath \mid \imath \in \{1,\ldots,k\}\}$, donde el número $\alpha_\imath$ es un factor de contracción para $f_\imath$, $\imath \in \{1,\ldots,k\}$.

De acuerdo con la Proposición 2.1.1, dado un sistemas de funciones iteradas $\{(X,d)\ ;\ f_1, f_2, \ldots, f_k\}$, se puede definir una contracción F en el espacio métrico completo $(\mathscr{H}(X),h)$, y por el Teorema del punto fijo existe un único $A_F \in \mathscr{H}(X)$, tal que éste es el punto fijo de la contracción, el conjunto A_F es llamado el conjunto fractal asociado al SFI y a F la contracción inducida por el SFI.

Ejemplo. Consideremos el siguiente SFI

$$\left\{(I^2, d_e)\ ; f_1, f_2, f_3)\right\},$$

donde

$$d_e \text{ es la distancia euclideana,}$$

$$I = [0,1],$$

$$f_1(x,y) := (1/2x + 1/4, 1/2y + 1/2),$$

$$f_2(x,y) := (1/2x, 1/2y),$$

$$f_3(x,y) := (1/2x + 1/2, 1/2y).$$

Si F es la contracción inducida por el SFI y tomamos el compacto C de $\mathbb{R}^2$ como la imagen siguiente.

Figura 1: Imagen que representa al compacto C de $\mathbb{R}^2$.

Las figuras que se muestran a continuación de izquierda a derecha y de arriba hacia abajo son respectivamente los resultados obtenidos para

$$F(C), F^{\circ(2)}(C) := F(F(C)), \ldots, F^{\circ(6)}(C).$$

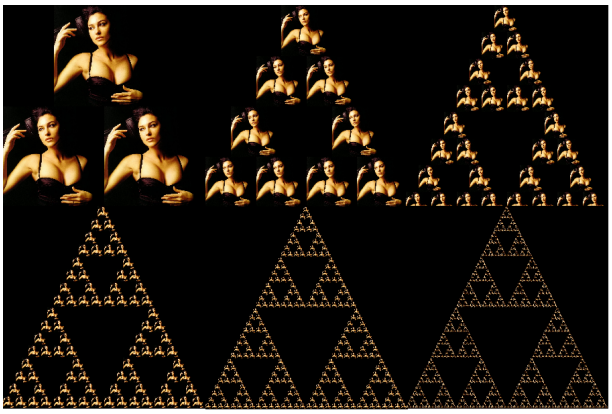


Figura 2: Como se puede abservar $F^{\circ(6)}(C)$ consta de 729 copias reducidas del compacto C.

Notemos que todo elemento de $(\mathscr{H}(X), h)$ se puede ver como el atractor de un SFI, pues dado $C \in (\mathscr{H}(X), h)$, la función

$$\phi : (\mathscr{H}(X), h) \longrightarrow (\mathscr{H}(X), h)$$

definida por, $\phi(A) := C$, para todo $A \in \mathscr{H}(X)$ es una contracción y

$$\{(\mathscr{H}(X), h) \; ; \; \phi\}$$

es un SFI, que tiene por atractor a C.

Este tipo de funciones son llamadas funciones de condensación[3] y a un SFI que contenga a una de estas funciones se le llama un sistema de funciones iteradas con condensación.

Teorema 2.1.3. ***(Teorema del collage para fractales).*** *Sea (X, d) un espacio métrico completo y sean $B \in \mathscr{H}(X)$ y $\varepsilon \in \mathbb{R}^+$ dados.*

Si $\{(X, d)\ ;\ f_1, \ldots, f_k\}$ es un SFI con un factor de contracción α y A_F el atractor del SFI tales que

$$h\left(B, \cup_{i=1}^{k} f_i(B)\right) \leq \varepsilon, \text{ entonces}$$

$$h(A_F, B) \leq \frac{\varepsilon}{1-\alpha}.$$

Es decir

$$h(A_F, B) \leq \frac{1}{1-\alpha} h\left(B, \cup_{i=1}^{k} f_i(B)\right),$$

para todo $B \in \mathscr{H}(X)$.

El Teorema del collage dice que para tener un SFI cuyo atractor sea semenjante a un conjunto $B \in \mathscr{H}(X)$, tenemos que fabricar un conjunto de contracciones $\{f_1, \ldots, f_k\}$, tal que la unión (o collage) de los conjuntos $f_1(B), \ldots, f_k(B)$ esté cercano al conjunto B.

Por lo tanto si $h\left(B, \cup_{i=1}^{k} f_i(B)\right) \leq \varepsilon$ para un ε lo suficientemente pequeño podemos sustituir a B por el atractor del SFI. El teorema también nos ayuda a tener una medida de que tan cerca estará el atractor de un SFI a B sin tener que calcular el atractor, basta sólo con estimar la distancia entre B y $\cup_{i=1}^{k} f_i(B)$.

Además la aproximación del atractor al conjunto B será mejor cuando más pequeño sea el factor de contración del SFI y no depende del número de contraciones que lo forman.

Si tomamos a $(\mathbb{R}^2, d_e)$, con d_e la métrica euclidiana y trabajamos sobre el conjunto de todos los subconjuntos compactos no vacíos de $\mathbb{R}^2$ para formar al espacio métrico $(\mathscr{H}(\mathbb{R}^2), h)$ en donde una fotografía o en general cualquier imagen es considerada como un compacto de $\mathbb{R}^2$, entonces podríamos aproximar una fotografía por un conjunto fractal que sea el atractor de un adecuado SFI y si además este SFI consta de pocas contracciones, podemos almacenar las contracciones en lugar de la imagen original y así habremos reducido el espacio ocupado por la imagen en la memoria de un sistema digital.

[3]Una función de condensación es una contracción y cero es un factor de contracción para ella.

Esta fue la idea que abrió la investigación de la compresión fractal de imágenes.

Existen algunos inconvenientes. Por desgracia el Teorema del collage no proporciona un método para encontrar dicho SFI y el SFI cuyo atractor aproxima al compacto en cuestión no necesariamente es único.

Si tenemos por ejemplo que $\{(X,d)\ ;\ \omega_1, \omega_2, \ldots, \omega_k\}$ con $k \geq 2$ es uno de los sistemas de funciones iteradas que aproxima a el compacto, entonces podemos escoger dos de las contracciones del SFI, digamos ω_1 y ω_2 para definir una nueva contracción $W_{1,2} : \mathscr{H}(X) \longrightarrow \mathscr{H}(X)$ como $W_{1,2}(A) := \omega_1(A) \cup \omega_2(A)$ para todo $A \in \mathscr{H}(X)$. Así tenemos un nuevo SFI

$$\{(\mathscr{H}(X), h)\ ;\ W_{1,2}, W_3 \ldots, W_k\}$$

donde

$$W_{1,2}(A) = \omega_1(A) \cup \omega_2(A) \text{ y } \forall\ i \in \{3, \ldots, k\}\,, \quad W_i(A) = \omega_i(A),$$

el cual tiene el mismo atractor que el SFI original y por tanto se acerca al compacto que queríamos aproximar del mismo modo, pero éste tiene $k-1$ contracciones a diferencia del primero que poseía k contracciones. De esta forma podemos definir diferentes SFI que tendrían el mismo atractor, sin embargo hay un aspecto importante que nos hacen preferir un SFI sobre otro y éste es que las contracciones que forman al SFI estén definidas de una manera simple lo cual nos permite intuir como ser el atractor del SFI. Cabe señalar que el Teorema del collage tampoco proporciona una manera de elegir el mejor SFI, sin embargo todo lo anterior no resta importancia a este resultado.

3 Sistemas de funciones iteradas por partes

El Teorema del collage asegura que dada cualquier imagen, siempre existe un fractal que se le parece tanto como nosotros queramos, pero supongamos que tenemos la fotografía de uno de nuestros seres queridos y que conocemos un SFI cuyo atractor aproxima a la fotografía, si iteramos la contracción inducida por el SFI evaluada en la imagen de un árbol, el atractor se parecerá a la fotografía que estamos intentando aproximar pero como está compuesta de copias transformadas de un árbol, la foto a detalle exhibiría pequeños arbolitos distorsionados, en particular en los rasgos físicos de la persona lo cual no es natural. Así los sistemas de funciones iteradas tienen este inconveniente. Hubo algunos intentos para resolver este problema pero no fue sino hasta 1989, cuando un estudiante de Michael F. Barnsley, Arnaud Jacquin diseño un nuevo método de codificación de imágenes basado en el

concepto fundamental de los SFI, pero haciendo a un lado el enfoque rígido de los SFI globales.

El nuevo método obedecía a una idea que en principio parece muy simple. En vez de ver a una imagen como una serie de copias transformadas de algún compacto arbitrario, esta vez la imagen estaría formada por copias de pedazos de si misma bajo transformaciones apropiadas. La mejilla de nuestra tía no se parece a un árbol pero es muy probable que su mejilla derecha si sea muy parecida a la izquierda, así pues esta idea contraresta en buena manera el problema que los SFI globales tienen. Este método se conoce como sistemas de funciones iteradas por partes SFIP.

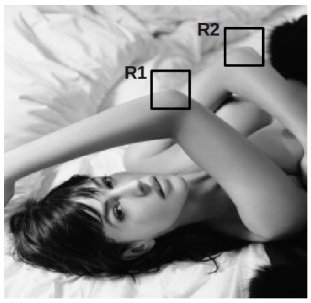


Figura 3: Existen varias partes de la imagen que se parecen entre si, en particular las que están encerradas en las regiones R1 y R2.

La idea general se basa en tomar una partición de la imagen, los elementos de la partición pueden tener formas arbitrarias. En [7], [14], [2] y [9] están descritos diversos algoritmos para realizar una partición, algunos ocupan triángulos o polígonos de tamaño variable, Éstos pueden mejorar la calidad de compresión pues facilitan encontrar la auto-semejanza entre las secciones de la imagen, pero también es posible que aumenten los costos de cómputo. La manera más sencilla de hacer la segmentación para una implementación es utilizando cuadrados de tamaño uniforme pues es más fácil delimitar los pixeles encerrados por estas regiones para realizar las comparaciones. Un estudio general del problema anterior se encuentra en [5].

Consideraremos a las imágenes como funciones que están definidas sobre I^2 y que tienen por contradominio a I, donde $I := [0, 1]$. A cada punto del cuadrado unitario le corresponde un valor que representa su nivel de gris, con esto podríamos establecer que si a un punto le corresponde el valor 0

en la imagen se representaría dicho valor como un punto negro, y el valor 1 se representaría en la imagen como un punto blanco.

Sea $A \subset I^2$, sabemos que una función $\varphi : A \longrightarrow I$ es un subconjunto de $A \times I$ con la propiedad de que para cada $(x, y) \in A$ existe un único $z \in I$, tal que $((x, y), z) \in \varphi \subset A \times I$ y generalmente a z se le denota como $\varphi(x, y)$. Por otra parte la gráfica de la función φ es un subconjunto de I^3 que consiste de todas las tripletas (x, y, z), tales que $x, y \in I$ y $z = \varphi(x, y)$, a la gráfica de una función φ la denotaremos por ${}^*(\varphi)$. Como puede observarse una función y su gráfica no son lo mismo, pero dada una función, su gráfica está implicitamente definida con ésta, y viceversa, si uno conoce la gráfica de una función puede de inmediato saber quién es la función. Como estaremos trabajando con las funciones antes mencionadas y sus gráficas, tomaremos la siguiente notación en adelante:

$$\mathscr{F} := \left\{ \varphi \subset A \times I \mid \varphi \text{ es una función y } A \subseteq I^2 \right\}$$

y definiremos a

$$\mathscr{G} := \left\{ {}^*(\varphi) \mid \varphi \in \mathscr{F} \right\}.$$

A cada elemento de $\mathscr{F}$ se le puede asociar de manera única un elemento de $\mathscr{G}$, recíprocamente a cada elemento de $\mathscr{G}$ se le puede asociar de manera única un elemento de $\mathscr{F}$, por tanto existe una biyección entre ambos conjuntos la función que definiremos a continuación es tal biyección.

$$\hbar : \mathscr{F} \longrightarrow \mathscr{G}$$

dada por

$$\forall\, \varphi \in \mathscr{F}; \quad \hbar(\varphi) := {}^*(\varphi).$$

La métrica del supremo la definiremos sobre nuestro conjunto $\mathscr{F}$ de manera usual como:

$$d_{sup} : \mathscr{F} \times \mathscr{F} \longrightarrow \mathbb{R}^+$$

dada por $\forall\, \varphi, \psi \in \mathscr{F}$

$$d_{sup}(\varphi, \psi) := sup \left\{ |z_1 - z_2| \mid x, y \in I, (x, y, z_1) \in {}^*(\varphi) \; y \; (x, y, z_2) \in {}^*(\psi) \right\}.$$

Proposición 3.1. *El espacio métrico $(\mathscr{F}, d_{sup})$ es un espacio métrico completo.*

Definición 3.2. Un sistema de funciones iteradas por partes o SFIP es una terna

$$\left(\left\{ (\mathscr{F}, d_{sup}); g_1, \ldots, g_n) \right\}, \mathscr{D}, \mathscr{R} \right)$$

formada por un SFI $\{(\mathscr{F}, d_{sup}); g_1, \ldots, g_n)\}$, una cubierta finita de I^2

$$\mathscr{D} := \{D_1, \ldots, D_k\},$$

tal que $\forall\, i \in \{1, \ldots, k\}$, $D_i \neq \emptyset$ y una partición de I^2

$$\mathscr{R} := \{R_1, \ldots R_n\},$$

tales que: Dada $\varphi \in \mathscr{F}$ y dada $i \in \{1, \ldots, n\}$, existen $\psi \in \mathscr{F}$ y $j \in \{1, \ldots, k\}$ tales que

$$g_i(\varphi|_{D_j}) = \psi|_{R_i}.$$

Definición 3.3. Se dice que una familia de funciones

$$\{g_i \mid g_i : \mathscr{G} \longrightarrow \mathscr{G}, i = 1, \ldots, n\}$$

enlosan[4] a I^2, si para toda $\varphi \in \mathscr{F}$, se tiene que

$$\bigcup_{i=1}^{n} g_i({}^*(\varphi)) \in \mathscr{G}.$$

Definición 3.4. Sea $f : A \subseteq \mathbb{R}^3 \longrightarrow \mathbb{R}^3$ una función y $f_1, f_2, f_3 : \mathbb{R} \longrightarrow \mathbb{R}$ sus funciones coordenadas, es decir

$$\forall\, (x, y, z) \in \mathbb{R}^3;\ f(x, y, z) = (f_1(x), f_2(y), f_3(z)),$$

entonces diremos que f es una función contraíble respecto a su tercer componente, si existe $\alpha \in [0, 1)$, tal que $\forall\, z_1, z_2 \in \mathbb{R}$,

$$d(f_3(z_1), f_3(z_2)) \leq \alpha\, d(z_1, z_2)$$

y $f_1(x), f_2(y)$ son independientes de z_1 y de z_2, para todo $x, y \in \mathbb{R}$. A α se le llama un factor de contracción respecto a la tercera componente.

Proposición 3.5. *Sean* $\left\{D_i \subseteq I^2 \mid i \in \{1, \ldots, n\}\right\}$ *una cubierta para* I^2,

$$\{f_i : \mathscr{G} \longrightarrow \mathscr{G} \mid i \in \{1, \ldots, n\}\}$$

una familia finita de funciones que enlosan I^2, *tales que*

$$\forall\, i \in \{1, \ldots, n\}, \forall\, \varphi \in \mathscr{F}; \qquad f_i({}^*(\varphi|_{D_i})) = f_i(x, y, \varphi(x, y)) \text{ con } x, y \in D_i$$

es una función contractiva respecto a su tercer componente y α_i *es un factor de contraccción. Entonces la función*

$$F : \mathscr{F} \longrightarrow \mathscr{F}$$

[4]La palabra enlosar significa cubrir un suelo con losas unidas y ordenadas. Por lo que preferimos la palabra enlosar en vez de cubrir, para enfatizar que pretendemos fabricar una cubierta enlosando la imagen con secciones (losas matemáticas) de sí misma.

dada por

$$\forall\ \varphi \in \mathscr{F}; \qquad F(\varphi) := \hbar^{-1}\left(\bigcup_{i=1}^{n} f_i(^*(\varphi))\right),$$

es una contracción en el espacio métrico $(\mathscr{F}, d_{sup})$ *y* $\alpha := máx\{\alpha_1, \ldots, \alpha_n\}$ *es un factor de contracción para* F*.*

Demostración. Recordemos que $\hbar$ es la biyección que existe entre $\mathscr{F}$ y $\mathscr{G}$.

Queremos probar que existe $\alpha \in [0, 1)$ tal que $\forall\ \varphi, \psi \in \mathscr{F}$, se cumple que:

$$d_{sup}(F(\varphi), F(\psi)) \leq \alpha\ d_{sup}(\varphi, \psi).$$

Sean $\varphi, \psi \in (\mathscr{F}, d_{sup})$y sea $\alpha = max\{\alpha_1, \ldots, \alpha_n\}$, estimemos la distancia entre $F(\varphi)$ y $F(\psi)$.

$d_{sup}(F(\varphi), F(\psi)) =$

$$= sup\{|z - w| \mid x, y \in I, (x, y, z) \in {}^*(F(\varphi)) \text{ y } (x, y, w) \in {}^*(F(\psi))\}$$

$$= sup\{|z - w| \mid x, y \in I, (x, y, z) \in {}^*(\hbar^{-1}(\cup_{i=1}^{n} f_i(^*(\varphi|_{D_i})))) \text{ y } (x, y, w) \in {}^*(\hbar^{-1}(\cup_{i=1}^{n} f_i(^*(\psi|_{D_i}))))\}$$

$$= sup\{|z - w| \mid x, y \in I, (x, y, z) \in {}^*(\hbar^{-1}(f_i(^*(\varphi|_{D_i})))) \text{ y } (x, y, w) \in {}^*(\hbar^{-1}(f_i(^*(\psi|_{D_i})))),\ i \in \{1, \ldots, n\}\}$$

$$= sup\{|z - w| \mid (x, y) \in D_i, (x, y, z) \in f_i(^*(\varphi|_{D_i})) \text{ y } (x, y, w) \in f_i(^*(\psi|_{D_i})),\ i \in \{1, \ldots, n\}\}$$

$$= sup\{|f_{i_3}(\varphi(x, y)) - f_{i_3}(\psi(x, y))| \mid (x, y) \in D_i, \varphi(x, y) = z \text{ y } \psi(x, y) = w,\ i \in \{1, \ldots, n\}\}.$$

Por la propiedad de contracción respecto a la tercera componente,$\forall\ i \in \{1, \ldots, n\}$ de f_i tenemos.

$$sup\{|f_{i_3}(\varphi(x, y)) - f_{i_3}(\psi(x, y))| \mid (x, y) \in D_i, \varphi(x, y) = z \text{ y } \psi(x, y) = w, \psi(x, y) = w,\ i \in \{1, \ldots, n\}\}$$

$$\leq sup\{\alpha_i\ |\varphi(x, y) - \psi(x, y)| \mid (x, y) \in D_i, \varphi(x, y) = z \text{ y } \psi(x, y) = w, i \in \{1, \ldots, n\}\}$$

$$\leq\ sup\{\alpha\ |\varphi(x, y) - \psi(x, y)| \mid (x, y) \in D_i, \varphi(x, y) = z \text{ y } \psi(x, y) = w, i \in \{1, \ldots, n\}\}$$

$$= \alpha \; sup\{|\varphi(x,y) - \psi(x,y)| \mid (x,y) \in I^2, \varphi(x,y) = z \text{ y } \psi(x,y) = w\}$$

$$= \alpha \; sup\{|z - w| \mid x, y \in I, (x,y,z) \in {}^*(\varphi) \text{ y } (x,y,w) \in {}^*(\psi)\}$$

$$= \alpha \; d_{sup}(\varphi, \psi),$$

por lo tanto

$$d_{sup}(F(\varphi), F(\psi)) \leq \alpha \; d_{sup}(\varphi, \psi).$$

Además $\alpha = \text{máx}\{\alpha_1, \ldots, \alpha_n\} \in [0, 1)$, así F es un contracción en $(\mathscr{F}, d_{sup})$ tal y como se quería demostrar. □

4 Implementación

El modelo matemático de una imagen como una función $\varphi : I^2 \longrightarrow I$ nos permite que la teoría funcione, sin embargo a la hora de tratar la información de una imagen con una computadora no necesariamente funcionará, pues hay que hacer la consideración de que en la computadora la función que modela a una imagen debe tener un dominio y rango discretos, de lo contrario una imagen representada por una función $\varphi : I^2 \longrightarrow I$ a pesar de tener un tamaño finito, describiría a una imagen de resolución infinita.

A lo que nos referimos en el párrafo anterior es que en una computadora se representa una imagen como una colección discreta de elementos de pigmento o pixeles, cada pixel toma un valor discreto en una escala de grises o bien tres en una escala con tres canales de color, el número de bits usados para almacenar estos valores es lo que se llama la resolución de la escala. Si usamos 8 bits para almacenar un solo valor por pixel, la imagen podría almacenarse en una computadora como una matriz de tamaño $n \times n$ donde cada entrada de la matriz sería un número entre 0 y 255, este valor representará un nivel en una escala de grises.

La imagen siguiente es un ejemplo de lo anterior, ésta tiene un total de 256×256 elementos de pigmento cada uno con un valor entre 0 y 255, para cada pixel se reservan 8 bits que almacenan su valor en binario. En total para poder almacenar la imagen en una computadora se ocuparían, 1 byte por pixel, lo que sería $256 \times 256 = 65\,536$ bytes o bien 64 kb.

Figura 4: Esta imagen es de 256 por 256 pixeles, la escala de grises es un valor entero entre 0 y 255, donde 0 es valor de negro y 255 el de blanco.

Ahora supongamos que dividimos esa misma imagen, en una cubierta $\mathscr{D}$ y una partición $\mathscr{R}$ ambas con elementos cuadrados de tamaño uniforme, los $D_i \in \mathscr{D}$ de tamaño 16×16 mientras que los $R_i \in \mathscr{R}$ sean de tamaño 8×8.

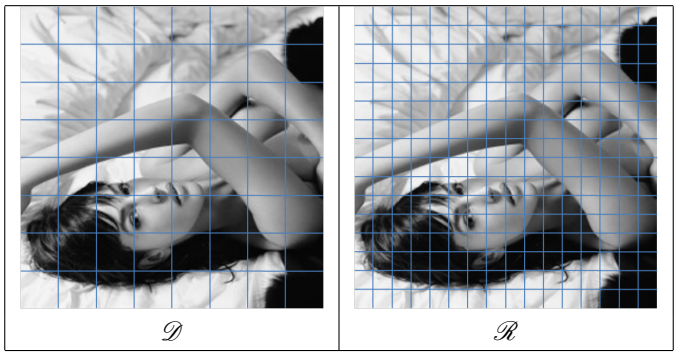


Figura 5: Segmentación de la imagen

De esta manera para cada uno de los $16 \times 16 = 256$ elementos $R_i \in \mathscr{R}$ que encierran 16 pixeles se tratará de encontrar un elemento $D_j \in \mathscr{D}$ de entre los $8 \times 8 = 64$ y una transformación tal que ésta evaluada en D_j minimice los valores en la escala de grises con respecto al R_i en cuestión, notemos que de entrada tanto los elementos de $\mathscr{D}$ como los de $\mathscr{R}$ son cuadrados con la particularidad de que lo primeros son del doble del tamaño de los segundos, la tranformación geométrica tendrá que ser por fuerza un escalamiento de un elemento de $\mathscr{D}$ a la mitad de su tamaño, y una traslación a la posición del elemento de $\mathscr{R}$ en turno, pero hay ocho posibles formas de mapear un

cuadrado en otro, cuatro rotaciones y cuatro reflexiones: vertical, horizontal y respecto a sus dos diagonales.

Para cada R_i, tal que $i = 1, \ldots, 256$ debemos tomar

$$\min_{k=1,\ldots,64} \left\{ \min_{j=1,\ldots,8} \{d(f_{ki}(\varphi(D_k)), \varphi(R_i))\} \right\}.$$

Lo anterior es costoso hablando en términos de cómputo ya que se requiere hacer para este caso un total de $8 \times 64 = 512$ comparaciones por cada elemento de $\mathscr{R}$ así que en resumen son $512 \times 256 = 131\ 072$ comparaciones.

El proceso de codificación no es muy complicado y se puede implementar con el siguiente algoritmo.

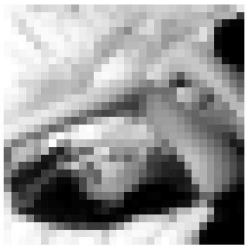
1. - Leer la imagen a ser codificada (traducirla en la matriz antes comentada).
2. - Segmentar la imagen en una cubierta $\mathscr{D}$ y una partición $\mathscr{R}$.
3. - Para un $R_0 \in \mathscr{R}$ dado, compararlo con cada una de las ocho posibles formas en que se puede mapear cada uno de los elementos de $\mathscr{D}$ en R_0. Obtener y almacenar el mejor $D_0 \in \mathscr{D}$ y la mejor transformación que aproximan a R_0.
4. - Repetir el paso anterior para cada uno de los elementos de nuestra partición $\mathscr{R}$.

El proceso de decodificación no es tan tardado comparado con el de codificado, como sabemos de la teoría desarrollada tenemos que iterar la función evaluada en una imagen cualquiera y obtener el punto fijo, el cual es el límite de esta sucesión de iteraciones. Veremos en los ejemplos que la sucesión se aproxima bastante rápido al punto fijo.

El algoritmo de decodificación es el siguiente:

1. - Leer los coeficientes de las funciones así como los D_i.
2. - Crear una imagen cualquiera del mismo tamaño de la imagen original.
3. - Tomar la cubierta $\mathscr{R}$ como en el paso de codificación, y a cada R_i aplicarle la transformación correspondiente.
4. - Hacer la imagen que resulta en el paso anterior la nueva entrada para el algoritmo hasta la iteración deseada.

Según el número de iteraciones la calidad de la imagen reconstruida va evolucionando para mejorar como puede observarse en la figura 6.



(a)

(b)

(c)

Figura 6: La figura 4 fue comprimida y en el proceso de reconstrucción resultan la imagen (a) haciendo una iteración, la imagen (b) al hacer cuatro iteraciones y (c) realizando ocho iteraciones.

Mostraremos más ejemplos como las imágenes 6 y 7 que fueron tomadas de [11] y las imágenes 8 y 9 de [12]. Para poder entender los resultados que se obtienen con los algoritmos anteriores al procesar estas imágenes necesitamos conocer lo que es el PSNR (Peek Signal to Noise Ratio), que es una de las formas más conocidas de medir en decibeles la calidad de una imagen reconstruida con respecto a una imagen original después de un proceso de compresión. También se requiere conocer el error cuadrático medio o MSE (Mean Square root Error). Supongamos que tenemos dos imágenes distintas I y F de tamaño $n \times m$, entonces las cantidades anteriores se definen de la siguiente manera:

$$MSE := \frac{1}{nm} \sum_{i=0}^{n-1} \sum_{j=0}^{m-1} ||I(i,j) - F(i,j)||^2,$$

$$PSNR := 10 \cdot log_{10}\left(\frac{MAX_F^2}{MSE}\right) = 20 \cdot log_{10}\left(\frac{MAX_F}{MSE}\right).$$

Donde MAX_F es el valor máximo que puede tomar un pixel en la imagen F.

Imagen original

Imagen reconstruida

Figura 7: La imagen original es de 512×512 pixeles, la imagen reconstruida tiene un PSNR = 30.00 dB y una razón de compresión de 70.29. Tiempo de codificado 1.7 segundos.

Imagen original

Imagen reconstruida

Figura 8: La imagen original es de 512×512 pixeles, la imagen reconstruida tiene un PSNR= 29.91 dB y una razón de compresión de 70.30. Tiempo de codificado 1.1 segundos.

Imagen original

Imagen reconstruida

Figura 9: La imagen original es de 512×512 pixeles, la imagen reconstruida tiene un PSNR = 33.19 dB y una razón de compresión de 7.93. Tiempo de codificado 2.12 segundos.

Imagen original

Imagen reconstruida

Figura 10: La imagen original es de 256×256 pixeles, la imagen reconstruida tiene un PSNR = 32.75 dB y una razón de compresión de 5.86. Tiempo de codificado 0.7 segundos.

5 Conclusiones

La técnica de compresión de imágenes digitales mediante conjuntos fractales generados por sistemas de funciones iteradas por partes es una ingeniosa aplicación de las matemáticas a la informática. Este trabajo sólo introduce

al lector en la implementación computacional de dicha técnica. Cabe mencionar que existen estudios en [5] en donde se trata el problema de mejorar el algoritmo en sus distintos pasos para lograr un desempeño más eficiente. Además de la compresión de imágenes, existen técnicas de super-resolución a diferentes escalas de una imagen, estas técnicas también se ayudan de la auto-semejanza o redundancia de la imagen [6].

Sergio Mabel Juárez Vázquez
CIDETEC,
IPN.
serge.galois@gmail.com

Flor de María Correa Romero
Departamento de Matemáticas,
Escuela Superior de Física y Matemáticas
IPN.
flor@esfm.ipn.mx

Referencias

[1] Barnsley M., Fractals Everywhere, Academic Press, New York 1988.

[2] Davione F.; Svensson J.; Chassery J., *A mixed triangular and quadrilateral partition for fractal image compression*, Proceedings of International Conference on Image Processing, Washington, DC (1995), No. **3**, 284–287.

[3] Ebrahimi M.; Vrscay E., *Self-similarity in imaging, 20 years after "Fractals Everywhere"*, Proceedings of International Workshop on Local and Non-Local Approximation in Image Processing, Lausanne (2008), 165–172.

[4] Falconer K., Fractal Geometry: Mathematical Foundations and Applications, John Wiley and Sons, 2003.

[5] Fisher Y., Fractal Image Compression: Theory and Application, Springer-Verlag, 1995.

[6] Glashner D.; Bagon S.; Irani M., *Super-resolution from a single image*, Proceedings of 12th International Conference on Computer Vision, 2009, 349–356.

[7] Harstenstein H.; Saupe D., *Cost-based region growing for fractal image compression*, Proceedings of IX European Signal Processing Conference, Rhodes 1998, 2313–2316.

[8] Hutchinson J.E., *Fractals and self-similarity*, Indiana Univ. Math. J. No. **30** (1981), 713–747.

[9] Kramm M., *Image cluster compression using partitioned iterated function systems and efficient inter-image similarity features*, Third International IEEE Conference on Signal-Image Technologies and Internet-Based System, Shanghai 2007, No. **1**, 989–996.

[10] Mandelbrot B., The Fractal Geometry of Nature, W.H. Freeman, 1977.

[11] Ochotta T. y Saupe D., *Edge-based partition coding for fractal image compression*, Arab. J. Sci. Eng., Special Issue on Fractal and Wavelet Methods (2004).

[12] Pérez J., *Codificación fractal de imágenes*, Universidad de Alicante, España, Tech. Rep., 1997. Una versión del documento está disponible en la dirección http://www.dlsi.ua.es/∼japerez/pub/pdf/mastertesi1998.pdf

[13] Rudin W., Real and Complex Analysis, MacGraw-Hill 1986.

[14] Saupe D.; Ruhl M., *Evolutionary fractal image compression*, IEEE International Conference on Image Processing, Lausanne 1996, 129–132.