HERRAMIENTA SOFTWARE PARA LA INTERPOLACIÓN DE DATOS EN 2D CON CURVAS AUTOAFINES NUNCA DIFERENCIABLES

GUSTAVO ADOLFO FONSECA MARTÍNEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERÍA DE SISTEMAS E INFORMÁTICA
BUCARAMANGA
2006

HERRAMIENTA SOFTWARE PARA LA INTERPOLACIÓN DE DATOS EN 2D CON CURVAS AUTOAFINES NUNCA DIFERENCIABLES

GUSTAVO ADOLFO FONSECA MARTÍNEZ

Trabajo presentado como requisito para optar al título de
Ingeniero de Sistemas

Director
Rafael Isaacs Giraldo

Codirector
Fernando Ruiz Díaz

UNIVERSIDAD INDUSTRIAL DE SANTANDER
ESCUELA DE INGENIERÍA DE SISTEMAS
BUCARAMANGA
2006

AGRADECIMIENTOS

A mi familia por su apoyo incondicional e infinita paciencia.

A Rafael Isaacs, director de este trabajo de grado por su paciencia, orientación y por compartir su peculiar punto de vista del mundo.

A mis compañeros y amigos por la ayuda desinteresada y por los momentos de diversión y esparcimiento. Quiero agradecer especialmente a Leonardo Fabio Padilla, Daniel Ardila, Juan Carlos Laverde, Jesús Adrian (Chucho), Sergio Gelvez, Oscar Murrillo, Samuel Hernández, Johatan Ramos Chauxx y a Odaimar Jesús Carrillo.

A Cesar Rivera, y Sergio Augusto Gálvez por su orientación y su forma de ver la vida.

Y sobretodo doy gracias a Dios por permitirme vivir en la forma en que deseo.

CONTENIDO

LISTA DE FIGURAS

RESUMEN

TITULO: HERRAMIENTA SOFTWARE PARA LA INTERPOLACIÓN DE DATOS EN 2D CON CURVAS AUTOAFINES NUNCA DIFERENCIABLES.*

AUTOR:
Gustavo Adolfo Fonseca Martínez**

PALABRAS CLAVE: Geometría fractal, interpolación de datos, ajuste de curvas, autosimilitud.

DESCRIPCIÓN:
Este trabajo de grado es un primer paso en el análisis y tratamiento de datos desde el enfoque de la teoría fractal trabajando la interpolación de datos en 2D. Siempre buscando aproximarse de una forma no convencional a la naturaleza de los fenómenos estudiados.
Gracias a la geometría Euclidiana, la trigonometría, y el cálculo estamos acostumbrados a modelar la realidad en términos de líneas rectas, polígonos, círculos, parábolas, y otros tipos de curvas simples. Esta forma de ver el mundo real es eficaz cuando modelamos objetos artificiales como paredes, automóviles y ciudades. Sin embargo cuando deseamos modelar sistemas naturales como los perfiles de una montaña, fenómenos económicos o sociales los componentes de estos sistemas no son bien descritos por funciones elementales o gráficos de funciones euclidianas.
Por lo expuesto anteriormente, se desarrollo una herramienta software para la interpolación de datos en 2D buscando una forma de comprensión mas completa de algunos fenómenos naturales, sociales o económicos.

---

* Trabajo de Grado.

** Facultad de Ingenierías Físico-Mecánicas, Ingeniería de Sistemas, Rafael Isaacs Giraldo.

ABSTRACT

TITLE: SOFTWARE TOOL FOR DATA INTERPOLATION IN 2D WITH AUTO-SIMILAR CURVES THAT ARE NEVER DIFFERENTIABLE[*].

AUTHOR: Gustavo Adolfo Fonseca Martínez[**].

KEY WORDS: Fractal geometry, interpolation of data, adjustment of curves, auto similarity.

DESCRIPTION: This graduation work is a first step in the analysis and data processing since the approach of the fractal theory working the data interpolation in 2D. It is always looking for an unconventional approach to the nature of the studied phenomena.

Thanks to the Euclidean geometry, trigonometry, and calculus we are accustomed to model the reality in terms of straight lines, polygons, circles, parabols, and other types of simple curves. This way of seeing the real world is effective when we are modeling artificial objects like walls, automobiles and cities. However, when we want to model natural systems like the cross section of a mountain, or the economic and social phenomena, the components of these systems are not well described by elementary functions or graphics of Euclidian functions.

Owing to the argument exposed previously, a software tool was developed for the data interpolation in 2D, searching for a more complete way of understanding some natural, social or economic phenomena.

---

[*] Degree Work

[**] Physical-Mechanical Faculty, Systems Engineering, Rafael Isaacs Giraldo.

INTRODUCCIÓN

Este trabajo de grado surge de la necesidad de introducir una nueva herramienta para la interpolación de datos en 2D ya que la mayoría de los métodos actuales se basan en las formas de la geometría euclidiana y el análisis clásico (generación de rectas, parábolas, funciones polinómicas, exponenciales, trigonométricas) guiados por el paradigma de lo diferenciable, lo cual no permiten acercarse de una forma eficaz a la disposición del conjunto de puntos de algunos fenómenos y de esta forma a la naturaleza del mismo. Enfocándonos en la autosimilitud y no en la diferenciabilidad se espera que el resultado de la interpolación permita predecir datos no observados y aportar a la compresión y comprensión global del fenómeno.

El primer paso fue la consecución de información acerca de la interpolación de datos en 2D, los métodos existentes basados en la geometría euclidiana no daban resultados cercanos a la realidad cuando se aplicaban a ciertos fenómenos naturales. Así fue como se determino que el objetivo del proyecto seria desarrollar una herramienta de interpolación de datos en 2D que permitiera un acercamiento al conjunto de datos observados desde el enfoque de la geometría fractal. Decidido esto el siguiente paso fue la documentación acerca de la geometría fractal, los sistemas iterativos de funciones y la autosimilitud.

Luego el estudio se enfoco en como la geometría fractal asocia diferentes maneras de generar modelos matemáticos que producen curvas que a primera vista parecen complicadas (en ningún punto diferenciables) esperando encontrar modelos que correspondan a fenómenos naturales, económicos o sociales.

En el capitulo uno se hace una presentación del proyecto así como el planteamiento del problema.

En el capitulo dos se presenta el marco teórico, los conceptos de interpolación y aproximación, los fundamentos de la geometría fractal, una descripción de los sistemas iterativos de funciones, el concepto de Codificación N-Proporcional y una descripción de la teoría de los algoritmos genéticos.

En el capitulo tres se explica cual fue la metodología de software utilizada en el desarrollo del proyecto.

En el cuarto capitulo corresponde al desarrollo del software, explicando cada una de las etapas que fueron necesarias para completarlo.

1. PRESENTACIÓN DEL PROYECTO

1.1 PLANTEAMIENTO DEL PROBLEMA

Actualmente se dispone de métodos de interpolación que permiten obtener una aproximación en cualquier punto del dominio. Sin embargo, estos métodos como el análisis de regresión, llegan a suavizar tanto los datos conocidos, que se pierde información; y otros, como unir los puntos con segmentos de recta, desprecian información que podría existir entre ellos. Esto produce que cualquiera de los dos métodos que se utilice para realizar la interpolación, ya sea regresión lineal, regresión cuadrática, o unión con segmentos de recta, ocasiona una pérdida de información en la deducción de una función que represente a los datos observados y por lo tanto, el pronóstico se ve afectado, pues resultan valores que no corresponden a la realidad.

La interpolación es ampliamente utilizada en la Ingeniería y ciencias al estudiar fenómenos de todo tipo. Gracias a la geometría Euclidiana, la trigonometría, y el cálculo estamos acostumbrados a modelar la realidad en términos de líneas rectas, polígonos, círculos, parábolas, y otros tipos de curvas simples es por eso que usualmente se utiliza cualquiera de los métodos de interpolación anteriormente mencionados. Esta forma de ver el mundo real es eficaz cuando modelamos objetos artificiales como paredes, automóviles y ciudades. Sin embargo cuando deseamos modelar sistemas naturales como los perfiles de una montaña, las cimas de las nubes, un copo de nieve o árboles, los componentes de estos sistemas no son bien descritos por funciones elementales o gráficos de funciones euclidianas es por esto que los métodos tradicionales de interpolación son imprecisos e insuficientes.

Por lo expuesto anteriormente, existe la necesidad de introducir una nueva herramienta para la interpolación de los datos observados. Ya que el resultado de la interpolación debe predecir datos no observados y aportar a la compresión y comprensión global del fenómeno.

1.2 OBJETIVOS

1.2.1 OBJETIVO GENERAL

Desarrollo de una herramienta software para interpolación de datos en 2D para el tratamiento y el análisis de datos bajo el enfoque de la geometría fractal, basado en la teoría de los sistemas iterativos de funciones

1.2.2 OBJETIVOS ESPECÍFICOS

1. Explorar el tratamiento de datos. Trabajando la interpolación 2D desde el enfoque de la teoría fractal.

2. Profundizar en el análisis de datos por métodos no clásicos, al determinar en la grafica fractal segmentos donde la curva es una copia, reducida y apropiadamente escalada, de toda la grafica.

3. Estudiar métodos que permitan una compresión de datos del fenómeno estudiado, buscando entre la población de datos inicial una muestra o subpoblación de datos que posean un carácter representativo o descriptivo del fenómeno.

2. FUNDAMENTACIÓN TEÓRICA

2. FUNDAMENTOS DE INTERPOLACION

Matemáticamente el problema de la interpolación consiste en escoger una prolongación adecuada de una función definida en un conjunto discreto. Sea el conjunto de datos $\Delta = \{(x_0, y_0), (x_1, y_1), \ldots, (x_n, y_n)\}$ donde $x_0 < x_1 < \ldots < x_n$ son números reales. Se tiene entonces una función *f* con dominio finito y tal que $f(x_i) = y_i$ para cada *i* en $\{0,1,...,n\}$.

$$f(x_0) = y_0, \quad f(x_1) = y_1, \quad ..., \quad f(x_n) = y_n. \tag{2.1}$$

Se requiere construir una función F (*función de interpolación*) que pertenezca a una clase conocida *C* y que tome los mismos valores en los puntos de interpolación que f, es decir, tal que:

$$F(x_0) = y_0, \quad F(x_1) = y_1, \quad ..., \quad F(x_n) = y_n. \tag{2.2}$$

El conjunto *C* puede ser por ejemplo las funciones continuas definidas en el intervalo $[X_0, X_n]$, también podemos pedir que sean derivables etc. La función *F* es entonces una extensión de *f*.

Geométricamente, esto significa que ha de hallarse una curva $y = F(x)$ de un cierto tipo específico que pase por el conjunto de puntos dado $M_i(x_i, y_i)$ $\{i = 0,1,2,...\}$. Ver Figura No. 1.

La fórmula de interpolación resultante $y = F(x)$ se utiliza ordinariamente para aproximar los valores de la función dada $f(x)$ para valores del argumento x que difieran de los puntos de interpolación. Cuando $x \in [x_0, x_n]$ esta operación se denomina interpolación de la función f y cuando $x \notin [x_0, x_n]$ se denomina extrapolación de la función f.

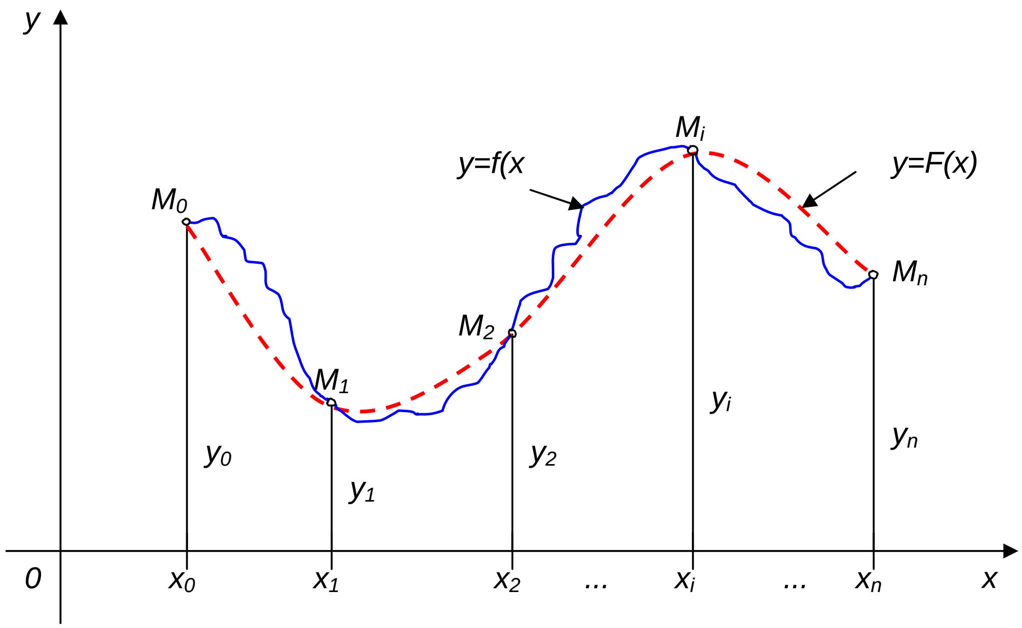

Figura 1. Representación de una función de interpolación.
Fuente: Cálculo Numérico Fundamental

Entre de los métodos utilizados para la interpolación de funciones se encuentran la Interpolación Lineal, Interpolación de Lagrange, Interpolación de Newton, etc.

2.1 GEOMETRIA FRACTAL

"... la *geometría fractal* no distingue, a propósito, entre conjuntos matemáticos (la teoría) y objetos naturales (la realidad). Incomparablemente más afín al mundo físico que la geometría euclidiana."

Mandelbrot

La geometría fractal es una teoría matemática reciente que se aparta radicalmente de la tradicional geometría Euclidiana, describe objetos que son autosimilares o simétricos en un factor de escala. Esto quiere decir que tales objetos cuando son magnificados, sus partes parecen ser una resemblanza del todo. Los Fractales no están relegados exclusivamente al mundo de las matemáticas. Si ampliamos un poco la definición de fractal, podremos encontrarlos virtualmente en cualquier lugar en el mundo natural. La diferencia es que los fractales "naturales" son randomicos, estadísticos, o estocásticos, más que exactamente simétricamente escalados[1].
Ya sean naturales o matemáticos, todos los fractales tienen una dimensión fractal particular. Estas dimensiones no son las mismas que las familiares dimensiones Euclidianas, medidas en forma entera 1, 2 o 3, son de un tipo diferente, usualmente no enteras, la dimensión fractal indica cuanto espacio ocupa el objeto dentro de la dimensión Euclidiana en la cual esta imbuido[2].

2.2 AUTOSIMILITUD

"Las cosas de incalculable complejidad se llaman *fractales* y tienen en común presentar longitudes infinitas dentro de áreas finitas."
Antonio Escohotado

Se refiere a la característica que presentan determinados objetos en los cuales los detalles mas pequeños que lo componen tiene alguna relación estadística con sus propiedades globales, repitiéndose tales detalles de una manera infinita (se refiere a una figura que es igual a sus partes salvo por un factor de escala). Un Fractal es, matemáticamente, una figura geométrica, compleja y detallada en estructura a cualquier nivel de magnificación. A menudo los fractales son semejantes a sí mismos; esto es, poseen la propiedad de que cada pequeña porción del fractal puede ser visualizada como una réplica a escala reducida del todo.

[1] Hayden White, *http://www.geometriafractal.com/articlech000300.htm*
[2] Hayden White, *http://pratt.edu/~arch543p/help/fractal_geometry.html*

2.3 SISTEMA ITERATIVO DE FUNCIONES (SIF)

Los Sistemas Iterativos de Funciones (SIF) sirven para formar conjuntos autosimilares, es decir aquellos conjuntos formados por partes iguales al "TODO", salvo por un factor de escala. La teoría de los SIF se desarrolla en el ambiente de los Espacios Métricos, tema profundamente trabajado por el Análisis Matemático del siglo XX.[3]

Definición 1: Un Sistema Iterativo de Funciones es una estructura $\{X, f_1, f_2, ..., f_n\}$ donde X es un espacio métrico completo (X, d) y cada $f_i : X \rightarrow X$ es una contracción. La notación de un SIF corresponde a $\{X\,;\, f_n, n = 1,2,...N\}$ y con factor de contracción $s = Max\{s_n : n = 1,2,..., N\}$, donde cada s_n es el factor de contracción de f_n, para $n = 1, 2, 3, ..., N$

Teorema 1: Sea $\{X, f_1, f_2, ..., f_n\}$ un SIF con factor de contracción s. Entonces la transformación $F : \mathrm{H}(X) \rightarrow \mathrm{H}(X)$ definida por :

$$F(K) = \bigcup_{i=1}^{n} f_i(K) \quad , \quad \forall K \in \mathrm{H}(X)$$

es una función de contracción sobre el espacio métrico completo $(\mathrm{H}(X), h)$ con factor de contracción s. Esto es:

$$h(F(K), F(C)) \leq s * h(K, C), \qquad \forall K, C \in \mathrm{H}(X)$$

---

[3] Nohora Isabel Najera, Interpolación Fractal: Analisis y Aplicación en 3D.

Por el teorema del punto fijo para espacios métricos completos para esta función F existe un *único punto fijo,* $A \in \mathrm{H}(X)$, tal que:

$$F(A) = A = \bigcup_{i=1}^{n} f_i(A)$$

Además para cualquier $K \in \mathrm{H}(X)$, se tiene que:

$$\lim_{n \to \infty} F^{\circ n}(K) = A$$

De otra forma: $\forall K \in \mathrm{H}(X)$

$$K, F(K), F^2(K), \ldots, F^n(K) \to A$$

Definición 2: El punto fijo $A \in \mathrm{H}(X)$ descrito en el teorema anterior es llamado el **atractor** del SIF. También se le llama conjunto **autosimilar** o **fractal**.

2.4 INTERPOLACION FRACTAL 2D

Los conceptos que a continuación se presentan son los que plantea Barnsley[4] y se consideran la base fundamental para el desarrollo de este proyecto.

Definición 3: Sea A un conjunto de puntos de la forma $\{(x_i, F_i) \in R^2 : i = 0,1,2,\ldots,N\}$, donde:

$x_0 < x_1 < x_2 < x_3 < \ldots < x_N$,

[4] BARNSLEY Michael, Fractals Everywhere, 1986.

Una función de Interpolación correspondiente a este conjunto de datos es una función continua $f:[x_{0,}x_N]\to R$ tal que:

$$f(x_i)=F_i \qquad \text{para} \quad i=0,1,2,...,N$$

Los puntos $(x_i, F_i)\in R^2$ son llamados los *puntos interpolados* y la función f es la función interpolante, la cual pasa a través de dichos puntos interpolados.

Sea el conjunto de datos $\{(x_i, F_i)\in R^2 : i=0,1,2,...,N\}$ dado, con el cual se puede construir un (SIF) en R^2 y obtener un atractor llamado G, el cual corresponde a la gráfica de una función continua $f:[x_{0,}x_n]\to R$, de datos interpolados.

Consideremos un SIF de la forma $(R^2, w_n, n=1,2,...,N)$, donde las funciones son transformaciones afines con una estructura especial, así:

$$w_n\begin{pmatrix} x \\ y \end{pmatrix}=\begin{pmatrix} a_n & 0 \\ c_n & d_n \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}+\begin{pmatrix} e_n \\ f_n \end{pmatrix}$$

Las transformaciones además son contracciones para los datos de acuerdo a:

$$w_n\begin{pmatrix} x_0 \\ F_0 \end{pmatrix}=\begin{pmatrix} x_{n-1} \\ F_{n-1} \end{pmatrix} \quad \text{y} \quad w_n\begin{pmatrix} x_N \\ F_N \end{pmatrix}=\begin{pmatrix} x_n \\ F_n \end{pmatrix} \quad \text{para} \quad n=1,2,3,...,N$$

Sea $n\in\{1,2,3,...,N\}$. Las transformaciones w_n están especificadas por cinco números reales a_n, c_n, d_n, e_n y f_n, las cuales se pueden expresar en las siguientes cuatro ecuaciones lineales :

$$a_n x_0 + e_n = x_{n-1}$$

$$a_n x_N + e_n = x_n$$

$$c_n x_0 + d_n F_0 + f_n = F_{n-1}$$

$c_n x_N + d_n F_N + f_n = F_n$

Se observa que en cada una de las transformaciones existe un parámetro libre (d_n). Al aplicar la transformación w_n, los segmentos verticales siguen quedando paralelos al eje y pero afectados por un factor $|d_n|$. Sea L un segmento de línea paralelo al eje y. Entonces $w_n(L)$ es también un segmento de línea paralelo al eje y. La relación de la longitud de $w_n(L)$ con la longitud de L *es* $|d_n|$. Llamamos a (d_n) el *factor de escala vertical* en la transformación w_n. Se escoge a d_n como un parámetro libre, para asignarle diferentes valores de escalado vertical. Por ejemplo si $d_n = 0$, y $n = 1,2,3,...,N$, se obtiene una función de interpolación lineal. Estos parámetros determinan la dimensión fractal de los atractores de los SIF.

Sea d_n un número real cualquiera. Demostramos que se puede siempre resolver las ecuaciones anteriores para a_n, c_n, d_n, e_n y f_n, en términos de los datos y de d_n, así

$$a_n = (x_n - x_{n-1})/(x_N - x_0) \quad (3.2.1)$$

$$e_n = (x_N x_{n-1} - x_0 x_n)/(x_N - x_0) \quad (3.2.2)$$

$$c_n = [(F_n - F_{n-1})/(x_N - x_0)] - [d_n (F_N - F_0)/(x_N - x_0)] \quad (3.2.3)$$

$$f_n = [(x_N F_{n-1} - x_0 F_n)/(x_N - x_0)] - [d_n (x_N F_0 - x_0 F_N)/(x_N - x_0)] \quad (3.2.4)$$

A continuación se muestra gráficamente como un SIF con sus respectivas transformaciones son utilizadas para construir una función de interpolación fractal. Ver Figura No. 2.

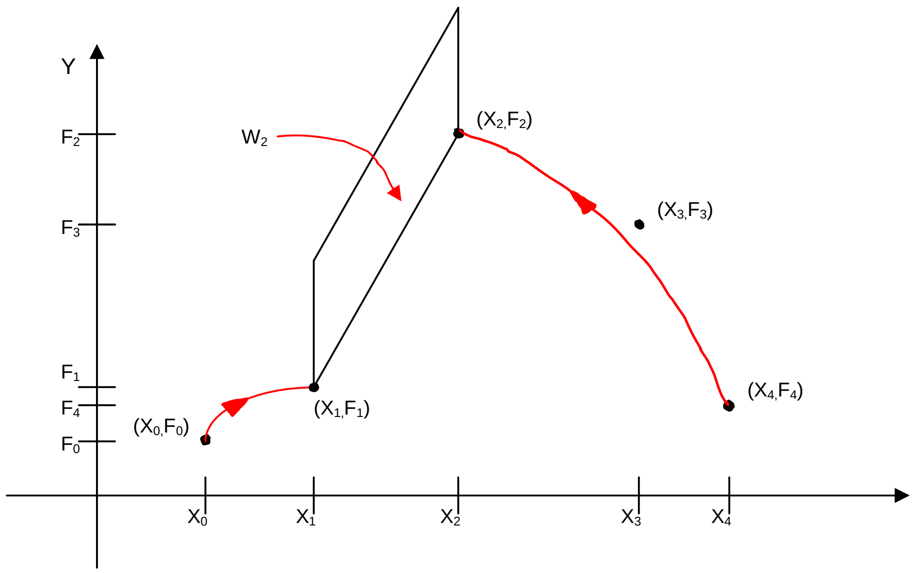

Figura 2. Construcción de una función de interpolación fractal a travéz de SIF

Fuente: Barnsley Michael. Fractals Everywhere, second edition.

Teorema 1: Sea N un entero positivo mayor que 1 y sea $\left(R^2, w_n, n=1,2,\ldots,N\right)$ un SIF como se definió anteriormente, asociado con el conjunto de datos $\{(x_n, F_n): n=1,2,3,\ldots,N\}$. Sea el factor de escala vertical d_n que cumple con $0 \leq d_n < 1$ para $n=1,2,3,\ldots,N$. Entonces hay una métrica d sobre R^2, equivalente a la métrica Euclidiana. En particular, hay un único conjunto compacto no vacío $G \subset R^2$ tal que

:

$$G = \bigcup_{n=1}^{N} w_n(G)$$

Teorema 2: Sea N un entero positivo mayor que 1 y sea $\left(R^2, w_n, n=1,2,\ldots,N\right)$ un SIF como se definió anteriormente, asociado con el conjunto de datos $\{(x_n, F_n): n=1,2,3,\ldots,N\}$. Sea el factor de escala vertical d_n que cumple con $0 \leq d_n < 1$ para $n=1,2,3,\ldots,N$, de modo que el SIF es hiperbólico. Sea G el atractor del SIF. Entonces G es la gráfica de una función continua $f:[x_{0,} x_n] \rightarrow R$ la cual contiene los datos interpolados $\{(x_i, F_i): i = 1,2,3,..., N\}$. Esto es:

$$G = \{(x, f(x)): x \in [x_0, x_N]\}$$

donde:

$$f(x_i) = F_i \qquad \text{para} \quad i =,1,2,3,..., N$$

Definición: La función f cuya gráfica es el atractor de un SIF como lo describe el Teorema 1 y 2 anteriores, es llamada una ***función de interpolación fractal*** correspondiente al conjunto de datos $\{(x_i, F_i): i = 1,2,3,..., N\}$.

En la Figura 3 se muestra un ejemplo de una sucesión de iteraciones $\{T^{on} f_0 :\ n = 0,1,2,3,...\}$ obtenidas por la aplicación repetida de funciones de contracción T, introducida en el Teorema 2. La función inicial $f_0(x)$ es lineal. La sucesión converge hacia una función de interpolación fractal f la cual es el punto fijo de T. Se puede observar que toda imagen puede ser interpretada como un atractor de un SIF con condensación, donde el conjunto condensación es la gráfica de una función f_0.

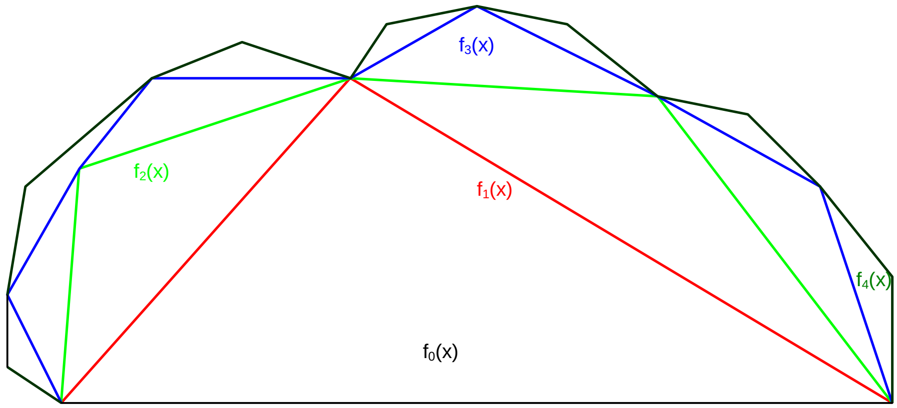

Figura 3. Representación de una función de contracción
Fuente: Barnlesy Michael. Fractals EveryWhere, second edition.

2.5 Codificación N-proporcional

La ***codificación N-proporcional*** aplicada a la interpolación de datos utiliza el principio de subdivisiones sucesivas, las cuales en este caso no son deben ser necesariamente en partes iguales. Dados $N+1$ números de la recta $x_0 < x_1 < x_2 < ... < x_N$ a cualquier x entre x_0 y x_N podemos asignarle un código (una sucesión de cifras cada una en el conjunto $\{0,1,...,N\}$) haciendo subdivisiones sucesivas cada una proporcional a la división que hacemos de los puntos $x_0, x_1, x_2, ..., x_N$ con el intervalo $[x_0, x_N]$[5].

[5] Nohora Isabel Najera, Interpolación Fractal: Analisis y Aplicación en 3D.

El algoritmo para la determinación de la base N-proporcional de un número es el siguiente:

Paso 1: Entran $x_0, x_1, x_2, ..., x_N$, Y (x)

Se hace:

$$P_i = \frac{x_i - x_{i-1}}{x_N - x_0} \quad \text{Para} \quad i = 1, 2, ..., N$$

Paso 2: Haga $t_0 = x_0$, ... , $t_N = x_N$

Paso 3: Para $j = 1, ..., c$

- Para $k = 0, 1, ..., N-1$:

  $$\text{Si} \quad x \in [t_k, t_{k+1}) \quad \text{entonces} \quad b_j = k$$

  Se toma el siguiente k. Se escoje la cifra $j - ésima$ según el intervalo al que x pertenezca.

- Se hace:

  $$t'_0 = t_k$$

  $$t'_{i+1} = t'_i + p_{i+1}\left(\frac{t_{i+1} - t_i}{t_N - t_0}\right)$$

- Se hace $t_i = t'_i$ para $i = 0, 1, ..., N$. (Subdivide el intervalo al cual pertenece x).
- Se repite el paso 3 y se toma el siguiente valor de j.

Utilizando el principio del la codificación N-proporcional se desarrollo el algoritmo para el cálculo puntual de la interpolación fractal, que consiste en dado un valor de x, conseguir su respectiva representación en la codificación N-proporcional y luego hallar su respectiva imagen. El algoritmo se presenta a continuación:

Paso 1: Entran $x_0, x_1, x_2, ..., x_N$ y x . Se hace:

$$P_i = \frac{x_i - x_{i-1}}{x_N - x_0} \quad \text{para} \quad i = 1, 2, ..., N$$

Paso 2: Se hace

$$t_0 = x_0 \quad ... \quad t_N = x_N$$

Paso 3: Para $j = 1, ..., c$

- Para $k = 0, 1, ..., N-1$:

  Si $x \in [t_k, t_{k+1})$ entonces $b_j = k$

  Se Toma el siguiente k. Se escoge la cifra $j-\acute{e}sima$ según el intervalo al que x pertenezca.

- Se hace:

$$t'_0 = t_k$$

$$t'_{i+1} = t'_i + p_{i+1}\left(\frac{t_{i+1} - t_i}{t_N - t_0}\right)$$

- Se hace $t_i = t'_i$ para $i = 0, 1, ..., N$. (Subdivide el intervalo al cual pertenece x).

- Se repite el paso 3 y se toma el siguiente valor de j .

Paso 4: Para $j = 1,...,n-1$

- Se dan los valores de d_j
- Se calculan los parámetros b , a_j , c_j , e_j , f_j
- Repetir el paso 4

Paso 5: Se inicializa x y y

Paso 6: Para $i = 1,...,c$

- Escoger aleatoriamente un número k comprendido entre 1 y $n-1$
- Calcular un nuevo valor de x

$$x' = a(k) * x + e(k)$$

- Calcular un nuevo valor de y

$$y' = c(k) * x + d(k) * y + f(k)$$

- Hacer $x = x'$ y $y = y'$
- Repetir el paso 6

Paso 7: Hacer $y_f = y$. El valor de y_f obtenido corresponde a la imagen del valor de x dado inicialmente.

2.6 ALGORITMOS GENETICOS

un algoritmo genético (AG) es una técnica de programación que imita a la evolución biológica como estrategia para resolver problemas. Dado un problema específico a resolver, la entrada del AG es un conjunto de soluciones potenciales a ese problema, codificadas de alguna manera, y una métrica llamada función de aptitud que permite evaluar cuantitativamente a cada candidata.

Estas candidatas pueden ser soluciones que ya se sabe que funcionan, con el objetivo de que el AG las mejore, pero se suelen generar aleatoriamente. Luego el AG evalúa cada candidata de acuerdo con la función de aptitud. En una población de candidatas generadas aleatoriamente, la mayoría no funcionarán en absoluto, y serán eliminadas. Sin embargo, por puro azar, unas pocas pueden ser prometedoras,

Estas candidatas prometedoras se conservan y se les permite reproducirse. Se realizan múltiples copias de ellas, pero las copias no son perfectas; se introducen cambios aleatorios durante el proceso de copia. Luego, esta descendencia digital prosigue con la siguiente generación, formando una nueva población de soluciones candidatas, y son sometidas a una ronda de evaluación de aptitud.

Las candidatas que han empeorado o no han mejorado con los cambios en su código son eliminadas de nuevo; pero, de nuevo, por puro azar, las variaciones aleatorias introducidas en la población pueden haber mejorado a algunos individuos, convirtiéndolos en mejores soluciones del problema, más completas o más eficientes. De nuevo, se seleccionan y copian estos individuos vencedores hacia la siguiente generación con cambios aleatorios, y el proceso se repite. Las expectativas son que la aptitud media de la población se incrementará en cada ronda y, por tanto, repitiendo este proceso cientos o miles de rondas, pueden descubrirse soluciones muy buenas del problema[6].

Para comenzar un AG se genera una población de forma aleatoria. El algoritmo genético procede de la forma siguiente:

1. Evaluar la puntuación (*fitness*) de cada uno de los individuos.
2. Permitir a cada uno de los individuos reproducirse, de acuerdo con su puntuación.
3. Emparejar los individuos de la nueva población, haciendo que intercambien material genético, y que alguno de los bits de un gen se vea alterado debido a una *mutación* espontánea.

Durante la evaluación, se halla la solución del problema a partir de esos parámetros, y se le da una puntuación a esa solución en función de lo cerca que esté de la mejor solución. A esta puntuación se le llama *fitness*.

El fitness determina siempre los individuos que se van a reproducir, y aquellos que se van a eliminar, pero hay varias formas de considerarlo para seleccionar la población de la siguiente generación:

1. Usar el orden, o rango, y hacer depender la probabilidad de permanencia o evaluación de la posición en el orden.

---

[6] http://the-geek.org/docs/algen/

2. Aplicar una operación al fitness para escalarlo; como por ejemplo el *escalado sigma.* En este esquema el fitness se escala
3. En algunos casos, el fitness no es una sola cantidad, sino diversos números, que tienen diferente consideración. Basta con que tal fitness forme un orden parcial, es decir, que se puedan comparar dos individuos y decir cuál de ellos es mejor. Esto suele suceder cuando se necesitan optimizar varios objetivos.

Una vez evaluado el fitness, se tiene que crear la nueva población teniendo en cuenta que los *buenos* rasgos de los mejores se transmitan a esta. Para ello, hay que seleccionar a los individuos encargados de esta tarea. Y esta selección, y la consiguiente reproducción, se puede hacer de tres formas principales:

- **Basado en el rango**: en este esquema se mantiene un porcentaje de la población, generalmente la mayoría, para la siguiente generación. Se coloca toda la población por orden de fitness, y los *M* menos dignos son eliminados y sustituidos por la descendencia de alguno de los *M* mejores con algún otro individuo de la población. A este esquema se le pueden aplicar otros criterios; por ejemplo, se crea la descendencia y esta sustituye al más parecido entre los individuos. Esto se denomina *crowding*, y fue introducido por **DeJong.** (Eliminar el individuo número 3, y se sustituirlo por un descendiente del número 2 y otro aleatorio, escogido entre el 1 y el 4). En realidad, para este esquema se escoge un *crowding factor, CF*. Cuando nace un nuevo individuo, se seleccionan CF individuos de la población, y se elimina al más parecido a la nueva criatura. Una variante de este es el muestreado estocástico universal, que trata de evitar que los individuos con más fitness copen la población; en vez de dar la vuelta a una ruleta con una ranura, da la vuelta a la ruleta con N ranuras, tantas como la población; de esta forma, la distribución estadística de descendientes en la nueva población es más parecida a la real.

- **Rueda de ruleta:** se crea un *pool* genético formado por cromosomas de la generación actual, en una cantidad proporcional a su fitness. Si la proporción hace que un individuo domine la población, se le aplica alguna operación de escalado. Dentro de este *pool*, se cogen parejas aleatorias de individuos y se emparejan, sin importar incluso que sean del mismo progenitor (para eso están otros operadores, como la mutación). Hay otras variantes: por ejemplo, en la nueva generación se puede incluir el mejor representante de la generación actual. En este caso, se denomina método *elitista*.

- **Selección de torneo:** se escogen aleatoriamente un número *T* de individuos de la población, y el que tiene puntuación mayor se reproduce, sustituyendo su descendencia al que tiene menor puntuación.

2.6.1 Crossover

Consiste en el intercambio de material genético entre dos o mas individuos. El *crossover* es el principal operador genético, hasta el punto que se puede decir que no es un algoritmo genético si no tiene *crossover*, y, sin embargo, puede serlo perfectamente sin mutación, según descubrió Holland. El *teorema de los esquemas* confía en él para hallar la mejor solución a un problema, combinando soluciones parciales.

Para aplicar el crossover, entrecruzamiento o recombinación, se escogen aleatoriamente dos miembros de la población. No pasa nada si se emparejan dos descendientes de los mismos padres; ello garantiza la perpetuación de un individuo con buena puntuación (y, además, algo parecido ocurre en la realidad; es una práctica utilizada, por ejemplo, en la cría de ganado, llamada *inbreeding*, y destinada a potenciar ciertas características frente a otras). Sin embargo, si esto sucede demasiado a menudo, puede crear problemas: toda la población puede aparecer dominada por los descendientes de algún individuo, que, además, puede tener caracteres no deseados, es uno de los principales problemas con los que se enfrentan los que aplican algoritmos genéticos.

El intercambio genético se puede llevar a cabo de muchas formas, pero hay dos grupos principales

***Crossover n-puntos*:** los dos cromosomas se cortan por n puntos, y el material genético situado entre ellos se intercambia. Lo más habitual es un crossover de un punto o de dos puntos.

***Crossover uniforme*:** se genera un patrón aleatorio de 1s y 0s, y se intercambian los bits de los dos cromosomas que coincidan donde hay un 1 en el patrón. O bien se genera un número aleatorio para cada bit, y si supera una determinada probabilidad se intercambia ese bit entre los dos cromosomas

***Crossover especializados*:** en algunos problemas, aplicar aleatoriamente el crossover da lugar a cromosomas que codifican soluciones inválidas; en este caso hay que aplicar el crossover de forma que genere siempre soluciones válidas. Un ejemplo de estos son los operadores de crossover usados en el problema del viajante.

2.6.2 Mutación

En la Evolución, una mutación es un suceso bastante poco común (sucede aproximadamente una de cada mil replicaciones), como ya se ha visto anteriormente. En la mayoría de los casos las mutaciones son letales, pero en promedio, contribuyen a la diversidad genética de la especie. En un algoritmo genético tendrán el mismo papel, y la misma frecuencia (es decir, muy baja).

Una vez establecida la frecuencia de mutación, por ejemplo, uno por mil, se examina cada bit de cada cadena cuando se vaya a crear la nueva criatura a partir de sus padres (normalmente se hace de forma simultánea al crossover). Si un número generado aleatoriamente está por debajo de esa probabilidad, se cambiará el bit (es decir, de 0 a 1 o de 1 a 0). Si no, se dejará como está. Dependiendo del número de individuos que haya y del número de bits por individuo, puede resultar que las mutaciones sean extremadamente raras en una sola generación.

No hace falta decir que no conviene abusar de la mutación. Es cierto que es un mecanismo generador de diversidad, y, por tanto, la solución cuando un algoritmo genético está estancado, pero también es cierto que reduce el algoritmo genético a una búsqueda aleatoria. Siempre es más conveniente usar otros mecanismos de generación de diversidad, como aumentar el tamaño de la población, o garantizar la aleatoriedad de la población inicial[7].

Un algoritmo genético tiene también una serie de parámetros que se tienen que fijar para cada ejecución, como los siguientes:

- **Tamaño de la población**: debe de ser suficiente para garantizar la diversidad de las soluciones, y, además, tiene que crecer más o menos con el número de bits del cromosoma, aunque nadie ha aclarado cómo tiene que hacerlo. Por supuesto, depende también del ordenador en el que se esté ejecutando.

- **Condición de terminación:** lo más habitual es que la condición de terminación sea la convergencia del algoritmo genético o un número prefijado de generaciones.

[7] http://geneura.ugr.es/~jmerelo/ie/ags.htm

3. METODOLOGÍA

La elección de la metodología de desarrollo va de acuerdo al tipo de proyecto a desarrollar e influye de manera decisiva en el éxito del proyecto. El modelo de ciclo de vida establece el orden en que se llevan a cabo las tareas y establece los criterios que determinan la secuencia o paso de una tarea a otra durante el desarrollo del software.

Para la selección de la metodología de desarrollo que se necesita en este proyecto de grado se deben tener en cuenta los siguientes requerimientos:

1. Algunos de los requerimientos iniciales pueden cambiar en el transcurso del proyecto.

2. Un prototipo inicial del software que contenga la parte central o núcleo del proyecto es necesaria en una etapa temprana del desarrollo.

3. Es recomendable tener reuniones frecuentes con los interesados en el proyecto.

4. Los interesados en el proyecto tienen una gran demanda de visibilidad del avance del proyecto.

Teniendo en cuenta las anteriores consideraciones se ha escogido al Prototipado Evolutivo (ver figura No. 4) como la metodología de desarrollo mas apropiada. Este modelo comienza con el desarrollo de un prototipo inicial que contiene los requerimientos más importantes del proyecto o núcleo central del mismo.

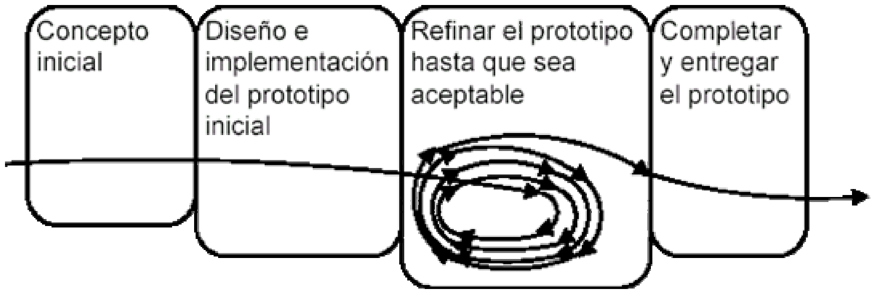

Figura 4. Modelo de prototipado evolutivo.
Fuente: Sotomayor Humberto. Método y técnicas para un desarrollo de software de calidad.

Esta metodología posee la ventaja de trabajar con base en la realimentación de requerimientos que se obtiene de la participación activa de los interesados en el proyecto en el proceso de desarrollo, esto da a lugar nuevos prototipos o incrementos en el software. Cada iteración de este tipo tiene como fin generar avances en el proyecto, este proceso se repite hasta que los requerimientos del proyecto o el tiempo establecido sean cumplidos.

El mayor problema de esta metodología de desarrollo es el desconocimiento tanto del tiempo necesario para terminar el proyecto como el número de iteraciones necesarias. Es por eso que se establece un periodo de tiempo aceptable en la entrega del producto, en el cual se debería cumplir con los objetivos específicos del proyecto.

3.1.1 ARQUITECTURA

La arquitectura del software alude a la estructura global del mismo y a las formas en que esta proporciona la integridad conceptual de un sistema. En su forma más simple, la arquitectura es la estructura jerárquica de los componentes o módulos del software, la manera en que interactúan (relación entre ellos) y la estructura de datos que van a utilizar los componentes.

Según la aplicación a desarrollar se escoge el modelo de arquitectura que se considere mas apropiada. Para el presente proyecto se escogió la arquitectura de capas, en concreto la arquitectura de tres capas (ver figura No .5).

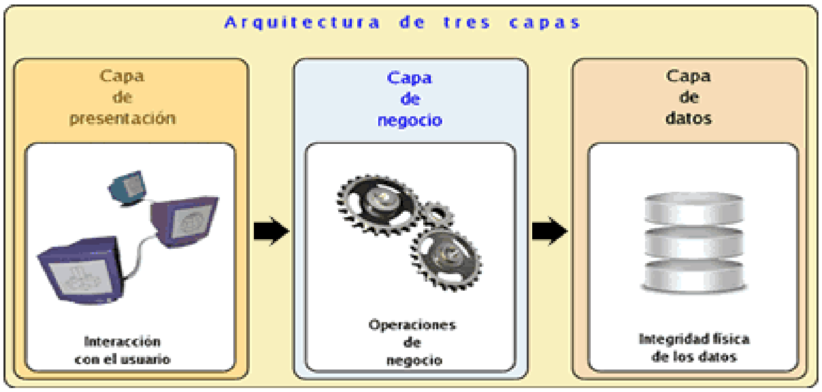

Figura 5. Arquitectura de tres capas.

1. **Capa de presentación:** Esta capa es la encargada de interactuar directamente con el usuario, por medio de la interfaz del programa.
2. **Capa de negocio:** Esta capa se encarga de realizar los procesos sobre los datos según las instrucciones del usuario.
3. **Capa de datos:** Almacena, recupera y mantiene la información necesaria para la ejecución de la aplicación.

4. DESARROLLO DEL SOFTWARE

4.1 FASE 1: CONCEPTO INICIAL (ENTENDIMIENTO DEL PROBLEMA)

Esta fase del proyecto consistió en las siguientes etapas:

1. Acercamiento a métodos de interpolación en 2D.
2. Documentación acerca de la geometría fractal.
3. Estudio de la interpolación fractal.

4.1.1 Acercamiento a métodos de Interpolación en 2D

Este primer paso fue un acercamiento a algunos de los métodos de interpolación en 2D existentes, para conocer la forma como se aproximaban a una distribución de puntos de ciertos fenómenos. Se encontró que los métodos actuales al basarse en las formas de la geometría euclidiana y el análisis clásico (generación de rectas, parábolas, funciones polinómicas, exponenciales, trigonométricas) son en la mayoría de los casos eficaces cuando tratamos datos provenientes de un sistema creado por el hombre (un sistema artificial), pero en algunos sistemas naturales (perfiles de montañas, nubes, costas de una playa).no se acercaban de una forma eficaz a una disposición de puntos.

4.1.2 Documentación acerca de la Geometría Fractal

En este paso se inicio la documentación acerca de la geometría fractal con un estudio acerca de la teoría de los sistemas iterativos de funciones y los Espacios Métricos que son el ambiente en los que se desarrollan. Luego se estudio el concepto de autosimilitud y su relación con los SIF.

4.1.3 Estudio de la Interpolación Fractal

En el ultimo paso de esta fase se estudio los conceptos de interpolación fractal descritos por Barnsley[8], esperando que al utilizar la teoría de los fractales en la interpolación se encontraran funciones que correspondan mejor a la disposición de puntos.

4.2 FASE 2: DISEÑO E IMPLEMENTACIÓN DEL PROTOTIPO INICIAL (REQUERIMIENTOS INICIALES)

En esta fase el primer paso fue realizar un diseño general de las funciones que incluiría la herramienta software a desarrollar lo cual sirvió para obtener un concepto global de la herramienta.

Establecidas las funciones de la herramienta software, se procedió al desarrollo del prototipo inicial el cual debería cumplir con las funciones predefinidas como fundamentales en el proyecto:

1. **Manejo de datos iniciales o conjunto de puntos iniciales junto con los parámetros de escala vertical d[ ]:** Se desarrollo inicialmente un formulario donde se pudiera escribir, modificar y borrar los parámetros de escala vertical d[ ], junto con los datos o puntos iniciales del sistema que van a ser interpolados.

2. **Construcción del SIF:** Una vez introducidos los datos y lo parámetros d[ ] se procede a la creación de las transformaciones w_n y posteriormente a la construcción del SIF.

3. **Grafica de la interpolación por segmentos o por lluvia de puntos:** Después de la creación del SIF la grafica de la interpolación puede ser

[8] BARNSLEY Michael, Fractals Everywhere, segunda edición, 1993.

visualizada de dos formas, por lluvia de puntos o por segmentos de recta que unen estos mismos puntos. Utilizando el algoritmo del cálculo puntual de la interpolación fractal se hallan valores al azar de (x) para luego determinar su respectiva imagen y de esta forma crear los puntos a graficar. Que tan elaborada se desee la grafica puede ser tan elaborada como se desee esto se logra fijando el número de iteraciones que se desee.

El desarrollo del prototipo inicial se dio por terminado en el momento que se cumplió con las tareas mencionadas.

4.3 FASE 3: REFINAR EL PROTOTIPO HASTA QUE SEA ACEPTABLE

Lo primero fue mostrar el prototipo inicial a los interesados por el proyecto para su evaluación. En esa entrevista al evaluar el prototipo surgieron algunas correcciones o mejoras al diseño, así como nuevos requerimientos o funciones que se deseaban en la herramienta.

En la siguiente parte de esta fase, comenzó con el refinamiento al primer prototipo, buscando corregir las fallas encontradas, mejorando el diseño y empezando el desarrollo de las nuevas funciones planteadas en la evaluación previa.

4.3.1 Correcciones y Mejoras

Grafica de la interpolación: La grafica de la interpolación se realizaba por medio de una lluvia de puntos o por segmentos de recta que unen estos mismos puntos. Se encontró que los puntos no estaban igualmente distribuidos a lo largo del plano XY, lo cual provocaba en el caso de la lluvia de puntos algunos segmentos de la grafica se vieran muy poblados en comparación de otros (ver figura No. 6), y en el caso de los segmentos de recta, debido al mismo problema de distribución de puntos algunas partes de la grafica parecían menos elaboradas que otras.

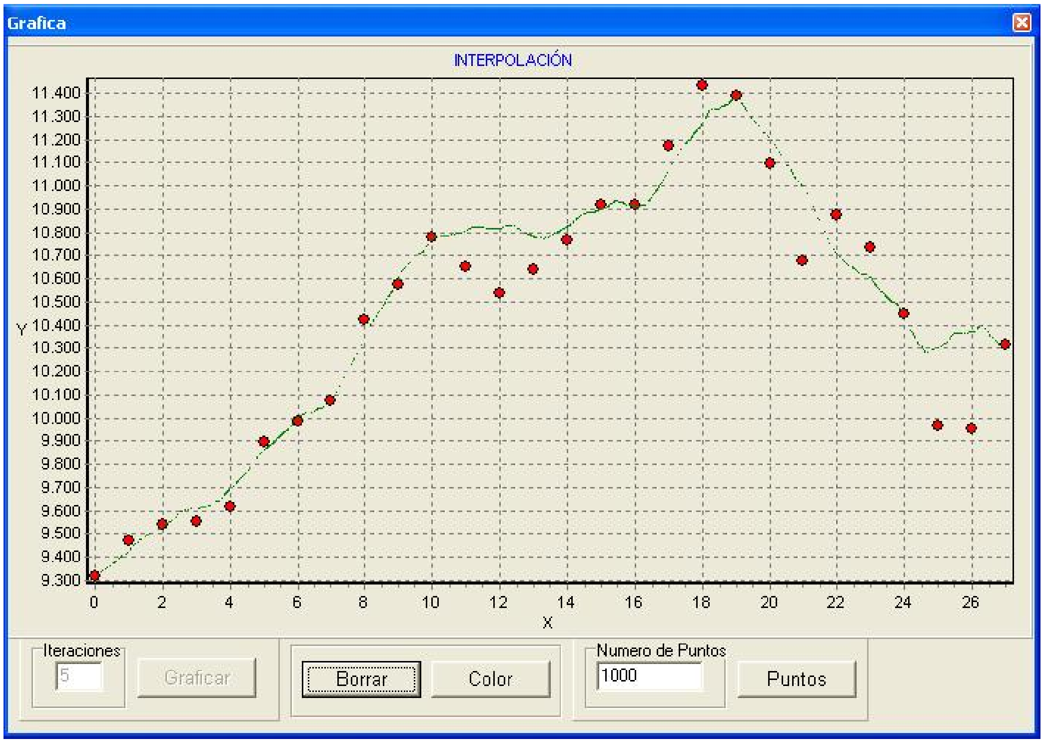

Figura 6. Grafica de interpolación por lluvia de puntos

- **Grafica recorriendo código:** La solución que se planteo consistió en asegurar que los puntos a graficar estuvieran uniformemente distribuidos, se ideo que partiendo del dato inicial (X[0]), se aumentara en un delta (x) hasta el dato final (X[final]) para luego utilizando el algoritmo del cálculo puntual de la interpolación fractal determinar su respectiva imagen (ver figura No 7).

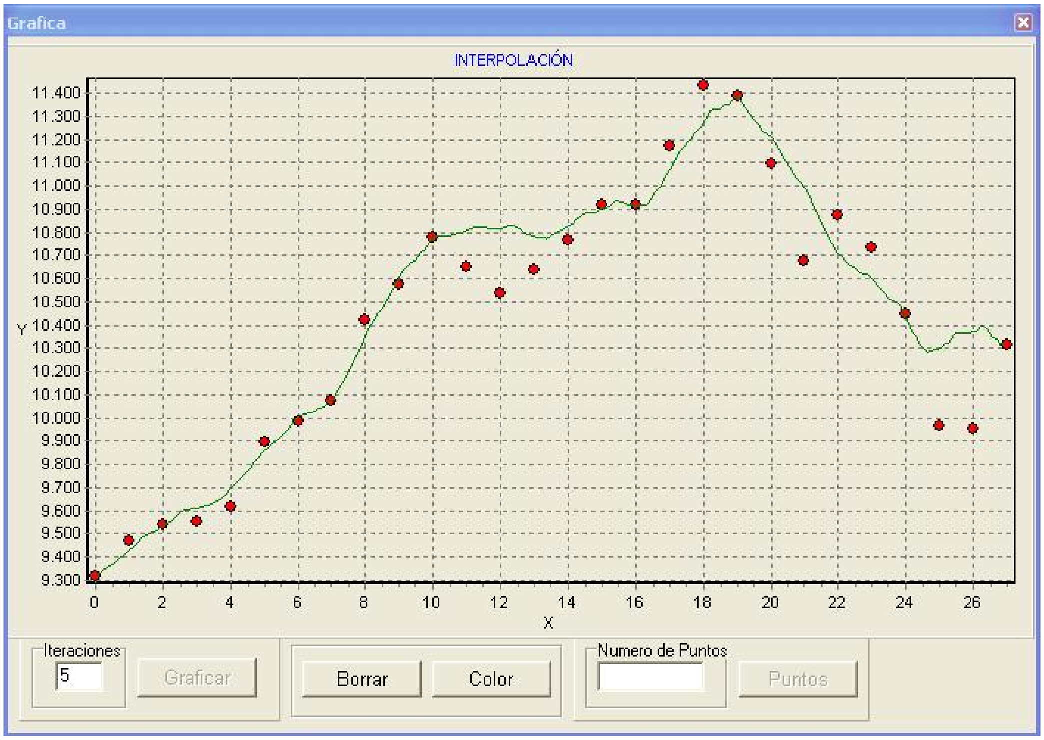

Figura 7. Grafica Interpolación recorriendo códigos.

Escoger el conjunto de puntos o datos iniciales: En el prototipo inicial se introducían únicamente los puntos a interpolar, se deseo la posibilidad de introducir una población de puntos iniciales para luego escoger entre ellos los puntos que serian interpolados (llamados pivotes) para luego crear las transformaciones y construir el SIF.

- **Seleccionar y cargar una población de puntos o datos iniciales:** Se acordó que la población de puntos o datos iniciales debían estar almacenados en archivos para transpórtalos o manejarlos fácilmente. Una vez cargado el archivo los datos pueden ser modificados en la ventana de edición para luego ser guardados (ver figura No.8).

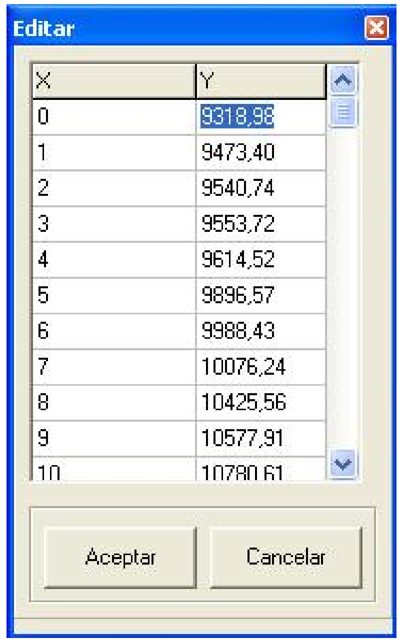

Figura 8. Ventana Editar Datos.

- **Escoger los pivotes o puntos a ser interpolados:** De la población inicial de puntos, se selecciona solo aquellos que se deseen interpolar (ver figura No. 9).

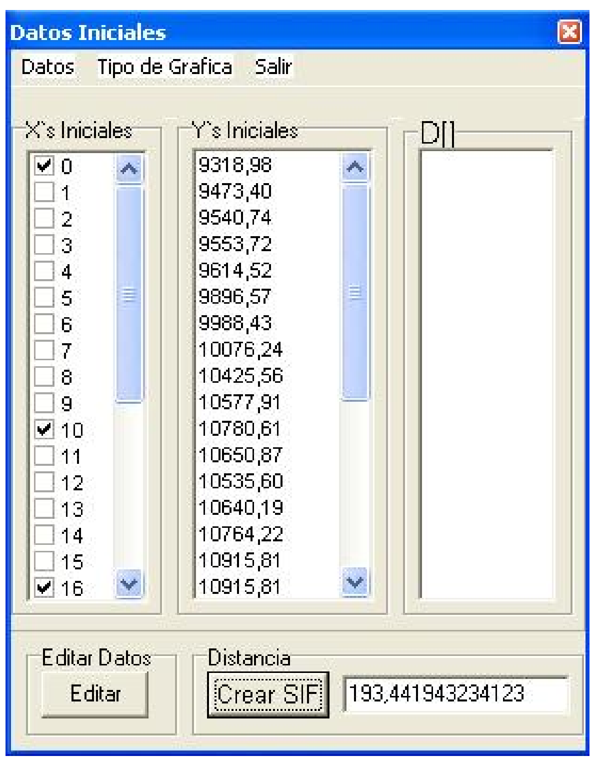

Figura 9. Ventana de Datos Iniciales.

Nuevos Requerimientos

Determinar los parámetros de escala vertical d[ ]`s y Construcción del SIF: Los parámetros de escala vertical determinan la dimensión fractal de los atractores del SIF. Es por eso que cuando se varían estos parámetros se logra una mayor o menor aproximación a la distribución de puntos iniciales, Se desea hacer un estudio buscando encontrar un algoritmo que encuentre los mejores valores para los parámetros de escala vertical (d[ ]).

La búsqueda del algoritmo para encontrar los mejore valores de los parámetros d[ ], resulto ser dentro de los nuevos requerimientos el mas importantes del proyecto. Es por eso que su evolución es descrita detalladamente a continuación

Paso 1 Función de distancia: El primer paso consintió en buscar una forma de determinar cuan cerca estaba un SIF de la distribución de puntos, por eso se creo la función distancia definida como:

$D = \sum_{n=0}^{N-1} \left(\sqrt{(Y(n) - y(n))^2} \right) / N$, Donde:

N: Número de datos.

Y(n): Es el valor de la coordenada Y de cada uno de los puntos de la población inicial o datos cargados desde el archivo.

y(n): Es el valor de la imagen de cada una de los datos X hallada.por el algoritmo del cálculo puntual de la interpolación fractal.

El promedio de las distancias que exista entre los dos valores me indica en que medida se aproxima un SIF a la distribución de puntos. Al variar los parámetros de escala vertical, cambia el valor de la distancia lo cual determina si se esta o no logrando aproximarse mejor a los datos.

Paso 2 del Algoritmo (Búsqueda al azar): Esta es la idea inicial del algoritmo, una vez escogido los pivotes:

1. Definir los valores de los parámetros d[ ] y crear las transformaciones.
2. Construir el SIF y revisar el valor de la función distancia.
3. Variar los valores de los parámetros d[ ], crear las transformaciones.
4. Construir el SIF y revisar nuevamente el valor de la función distancia..
5. Se escoge los valores para los parámetros d[ ] que den como resultado el menor valor de la función distancia

De esta forma es posible determinar cual de los dos valores para los parámetros d[ ] son mejores. Pero hacer este proceso manualmente seria literalmente eterno y posiblemente infructuoso. Es por eso que se escogió el camino de los métodos computacionales como el camino a seguir.

Se pensó incluir en el algoritmo una busqueda al azar de los valores de los parámetros d[ ]. Siguiendo con esta idea el algoritmo cambio de esta forma.

1. Definir los valores de los parámetros d[ ] con valores al azar (randomicos), crear las transformaciones.
2. Construir el SIF y revisar el valor de la función distancia.
3. Repetir los paso 1 y 2 el número de veces que se desee para crear una población de SIF (se recomienda una población de cien individuios).
4. Revisar el valor de la función distancia de cada uno de los cien SIF.
5. Ordenar de menor a mayor los SIF teniendo el valor de la función distancia como parámetro de selección.
6. Mostrar los mejores valores encontrados.

Paso 3 del Algoritmo (Búsqueda por algoritmos genéticos): Los resultados de paso anterior mostraron que el método de búsqueda al azar por si solo era insuficiente, se necesitaba otro método para lograr una mayor aproximación.

Se pensó en los algoritmos genéticos como una opción. Se continuo la idea paso anterior.

1. Definir los valores de los parámetros d[ ] con valores al azar (randomicos), crear las transformaciones.
2. Construir el SIF y revisar el valor de la función distancia.
3. Repetir los paso 1 y 2 el número de veces necesario para crear una nueva población de SIF.
4. Revisar el valor de la función distancia de cada uno de los cien SIF, la función distancia representa nuestro parámetro *fitness*.en el AG.
5. Ordenar de menor a mayor los SIF teniendo el valor de la función distancia como parámetro de selección.
6. Mostrar el mejor valor encontrado para la función distancia y los valores de los parámetros d[ ] que lo obtienen.
7. Se determinan dos valores de rangos dentro de la población de SIF de entre los cuales se obtendrán los dos padres necesarios para el entrecruzamiento y obtener un hijo.
8. Se repite el paso 8 el número de veces necesario hasta crear una nueva población de SIF, luego se repiten los pasos 4, 5, 6 este mismo número de veces.

La función de entrecruzamiento esta definida de la forma:

$$:\sum_{n=1}^{N} Hijo(d[n]) = \sum_{n=1}^{N} \left[padre1(d[n]) + padre2(d[n]) / 2 \right]$$, Donde:

N: es el número de parámetros = numero de pivotes -1.

9. Se le aplicara la función mutación a cada uno de estos individuos (SIF) la cual aplica la función MUTAR a cada uno de los parámetros d[ ] para crear nuevos individuos, luego se repiten los pasos 4, 5, 6.

La función mutación esta definida:

$$\sum_{n=1}^{N} SIF(d[n]) = \sum_{n=1}^{N} MUTAR(SIF(d[n]))$$

La función MUTAR después de varias pruebas quedo finalmente definida:

$\mathrm{MUTAR} = \mathrm{random}(\mathrm{deltaM},\text{-}\mathrm{deltaM}) + (d[n])$, Donde:

random: Regresa un valor dentro del intervalo $(\mathrm{deltaM},\text{-}\mathrm{deltaM})$.

deltaM: Es un valor que se puede seleccionar y determina en que medida crece o decrece el valor del parámetro d[ ].

Paso 4 del Algoritmo (Entrecruzamiento por combinaciones convexas): Se introdujo la teoría de las combinaciones convexas al algoritmo dentro de la función de entrecruzamiento. En este caso ya no serán 2 padres, sino 4 los que entraran en acción para crear un nuevo individuo. La nueva función de entrecruzamiento quedo definida:

$$\sum_{n=1}^{N} Hijo(d[n]) = \sum_{n=1}^{N} [padre1(d[n]) * \alpha + padre2(d[n]) * \beta + padre3(d[n]) * \chi + padre4(d[n]) * \delta]$$

$\alpha = random.$
$\beta = (1 - \alpha) * random.$
$\chi = (1 - [\alpha + \beta]) * random.$
$\delta = 1 - (\alpha + \beta + \chi).$

Con esta nueva función de entrecruzamiento se creara una nueva población de individuos, la cual será ordenada para luego ser mutada, logrando el máximo de eficacia en la búsqueda de los valores de los parámetros d[ ] (Ver figura No. 10).

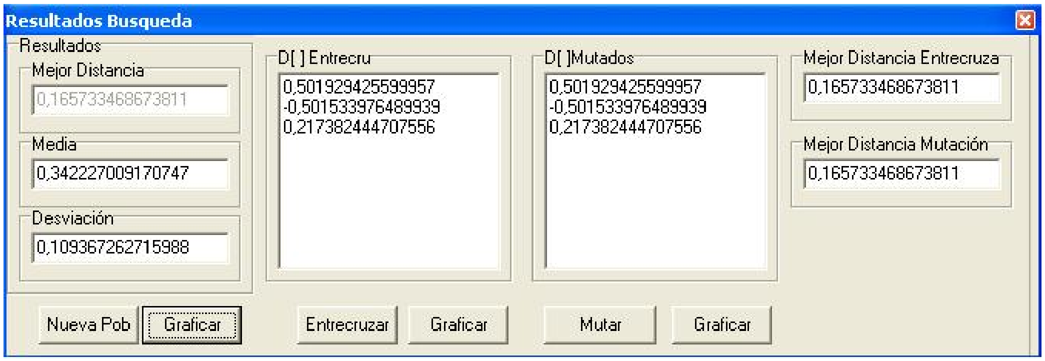

Figura 10. Ventana Resultados de la Búsqueda

4.4 FASE 4: COMPLETAR Y ENTREGAR EL PROTOTIPO

Esta fase comenzó cuando se cumplieron con las correcciones, mejoras y nuevas funciones mencionadas en la fase anterior. En este momento, todos los requisitos y objetivos específicos planteados inicialmente fueron cubiertos por la herramienta desarrollada. Las características de este producto final son:

- Permite seleccionar y cargar desde un archivo una población de puntos o datos iniciales. Estos archivos pueden ser modificados.
- Da la opción de escoger entre dos formas de visualizar la grafica de la interpolación.
- Permite seleccionar y modificar los parámetros necesarios para iniciar el algoritmo de búsqueda de los valores de los parámetros de escala vertical d[ ].

5. PRUEBAS Y RESULTADOS

Una vez terminado el software se llevaron a cabo diversas pruebas para descubrir posibles errores. En estas pruebas participaron tanto el desarrollador como los interesados de la herramienta para recoger diferentes opiniones respecto del sistema

Las pruebas son parte fundamental de un producto software antes de su entrega definitiva, con base en este concepto se llevaron a cabo tres tipos de pruebas:

- De tiempo de desarrollo
- De aceptación
- De validación

5.1 DE TIEMPO DE DESARROLLO

Este tipo de pruebas son aquellas que se realizan por parte de los desarrolladores informal y periódicamente, durante la etapa de desarrollo del software. Por ende, no tienen un orden definido. Estas pruebas permitieron verificar la funcionalidad de cada módulo y fueron realizadas durante toda la etapa de programación y empalme de los módulos que conforman el software.

5.2 DE ACEPTACIÓN

Son aquellas donde los interesados en el software comprueban la funcionalidad del sistema, y determina si acepta el software como está o precisa ser necesario aplicar nuevas optimizaciones y soluciones de fallas.

5.3 DE VALIDACIÓN

Al inicio de estas pruebas se usaron los módulos por separado durante la codificación para corroborar su correcto funcionamiento, y de esta forma facilitar al la prueba por parte del desarrollador, Luego se prueba el software totalmente ensamblado como un todo para comprobar si cumple los requisitos y funciones.

6. CONCLUSIONES

Desarrollo de una herramienta software para interpolación de datos en 2D para el tratamiento y el análisis de datos bajo el enfoque de la geometría fractal, basado en la teoría de los sistemas iterativos de funciones

- El uso de la teoría fractal para la interpolación de datos en 2D no solo es viable, también promete ser un campo de investigación rico aplicaciones para todas las ingenierías y ciencias.
- A pesar de que los SIF son una teoría poco difundida, presenta algunas ventajas sobre las demás teorías orientadas a lo diferenciable de las funciones interpolantes.
- Los el tratamiento de datos a través de los SIF abre nuevas puertas en el análisis de datos y amplia el espectro del investigador en este campo.
- Este tipo de proyecto de investigación requieren de un alto grado de compromiso entre el desarrollador y los interesados en el producto final
- La búsqueda de datos representativos o descriptivos de un fenómeno, mostró ser de gran complejidad.
- El estudio sobre los parámetros de escala vertical d[ ] mostró como se puede mejorar la aproximación de una curva a la disposición de datos sin variar los pivotes.

7. RECOMENDACIONES

- Continuar el mejoramiento del algoritmo de búsqueda de los parámetros de escala vertical d[ ].

- Desarrollar un modulo para controlar y manipular cualquiera de los SIF que sean creados en la ejecución de la herramienta.

- Crear un modulo de ayuda, para los posibles usuarios que no estén familiarizados con los términos o concepto de la teoría fractal o de los algoritmos genéticos.

8. BIBLIOGRAFÍA

Libros:

1. BARNSLEY, Michael, F., Fractals Everywhere, Segunda Edición, EEUU, 1993.

Tesis de grado:

1. NAJERA, Nohora Isabel "Interpolación Fractal: Analisis y Aplicación en 3D". Bucaramanga : UIS, 2003

Sitios web:

1. Teoría geometría fractal.
   http://www.geometriafractal.com/articlech000300.htm

2. Teoría geometría fractal.
   http://pratt.edu/~arch543p/help/fractal_geometry.html

3. Información, Algoritmos genéticos.
   http://geneura.urg.es/~jmerelo/ie/ags.htm

4. Información, Algoritmos genéticos.
   http://the-geek.org/docs/algen