E.T.S. de Ingeniería Industrial,
Informática y de Telecomunicación

Tratamiento de imágenes con ruido impulsivo mediante reglas difusas y algoritmos genéticos

Grado en Ingeniería Informática

Trabajo Fin de Grado

Sergio Sada Pezonaga

José Antonio Sanz Delgado

Pamplona, 25 de junio de 2015

upna Universidad Pública de Navarra
Nafarroako Unibertsitate Publikoa

RESUMEN

Uno de los principales problemas a resolver al realizar un procesamiento de imágenes es el tratamiento de imágenes con ruido en el que se tratará de eliminar la mayor cantidad posible de ruido.

Existen muchas técnicas para solucionar este problema. Una de ellas son los operadores Fuzzy Inference Ruled by Else-action (FIRE), enfocados a la eliminación de ruido impulsivo Sal y Pimienta.

Este algoritmo propone el uso de un algoritmo genético y la utilización de reglas difusas para tratar dicho problema. Una de las partes fundamentales del éxito de este algoritmo resida en la posibilidad de elección de los parámetros que modelan los conjuntos difusos.

En la propuesta original, los valores de estos parámetros son fijos por lo que puede acarrear que los resultados no sean todo lo buenos que pudieran ser.

Por lo tanto, el objetivo de este proyecto es proponer un nuevo algoritmo genético en el cual se aprenderá la mejor base de reglas y además, los parámetros que definan las funciones de pertenencia.

Para evaluar la calidad de la propuesta, evaluaremos y realizaremos pruebas en las que comprobaremos la nueva propuesta con el algoritmo original, además de la utilización de otras técnicas habitualmente utilizadas para resolver este problema.

Palabras clave

Tratamiento de imágenes, ruido sal y pimienta, conjuntos difusos, algoritmos genéticos, FIRE.

ÍNDICE

1. INTRODUCCIÓN

Procesamiento de imagen

El procesamiento digital de imágenes es un campo de investigación abierto. El constante progreso en esta área ha estado guiado por diferentes áreas como las matemáticas, la computación, así como el conocimiento cada vez mayor de ciertos órganos del cuerpo humano que intervienen en la percepción y en la manipulación de las imágenes. Unido a esto, la inquietud del hombre por imitar y usar ciertas características del ser humano como apoyo en la solución de problemas. El avance del procesamiento digital de imágenes se ve reflejado en la medicina, la astronomía, geología, microscopía, etc.

Una imagen digital es una representación bidimensional de una imagen en forma de una matriz numérica. En las imágenes representadas en escala de grises, los píxeles están representados con un valor numérico entero que está entre 0 y 255.

Uno de los principales problemas que se encuentran hoy en día es la aparición de ruido en imágenes digitales. Las principales fuentes de ruido aparecen durante las fases de adquisición y transmisión de imágenes. Existen muchos tipos de ruido y entre los más conocidos se encuentran algunos como el gaussiano o impulsivo.

Para la eliminación o minimización del ruido existen muchos métodos como el proceso de filtrado. Son un conjunto de técnicas cuyo objetivo principal es obtener, a partir de una imagen inicial con un determinado ruido, otra final cuyo resultado sea otra imagen con mejor calidad.

Los principales objetivos que se buscan con la aplicación de filtros son suavizar la imagen, eliminar el ruido, realce y detección de bordes.

Lógica difusa

Una de las disciplinas matemáticas con mayor número de investigadores en la actualidad es la llamada lógica difusa. Utiliza expresiones que no son ni totalmente ciertas ni completamente falsas, es decir, es la lógica aplicada a conceptos que pueden tomar un valor cualquiera de veracidad dentro de un conjunto de valores. La lógica difusa permite tratar información imprecisa como estatura media o temperatura baja.

El concepto de lógica difusa fue concebido por Lofti A. Zadeh en 1965. Zadeh estaba inconforme con los conjuntos clásicos que solo permitían dos opciones (la pertenencia o no de un elemento a dicho conjunto), y presentó la lógica difusa como una forma de procesar la información permitiendo pertenencias parciales a unos conjuntos.

La intención original del Zadeh fue la de crear un formalismo para manipular de forma más eficiente la imprecisión y la vaguedad del razonamiento humano expresado lingüísticamente. Sin embargo, causó cierta sorpresa que el éxito de la lógica difusa

llegase en el campo del control automático de procesos debido al tremendo interés causado en Japón allá por 1987.

Algoritmos genéticos

A lo largo de los años, el ser humano ha conseguido incrementar su capacidad de interactuación con entornos antes impensables y comprender la dificultad de muchos otros en los que no se ve tan capacitado. Por lo tanto, su habilidad para predecir la naturaleza del entorno cobra cada vez más importancia. Entre las ciencias que más han hecho por conseguir estos logros, la inteligencia artificial ocupa un lugar destacado.

En la década de los ochenta, el interés por la biología como fuente de inspiración para las ciencias de la computación comienza a ser una realidad entre los científicos, que vieron como este interés y su posterior desarrollo les ha llevado a logros nunca antes pensados. Logros que se han conseguido con desarrollos como las redes neuronales, el aprendizaje o la computación evolutiva y sus algoritmos genéticos.

Los algoritmos genéticos imitan el proceso evolutivo de la naturaleza para resolver problemas de optimización o búsqueda. En concreto, cada individuo en un algoritmo genético representa una solución al problema y sobrevivirán aquellos mejor adaptados, es decir, aquellos que ofrezcan una mejor solución al mismo. Para ello, se imitan mecanismos de la naturaleza como la selección, la reproducción o la mutación.

Objetivos

Los objetivos de este proyecto son varios. El objetivo inicial es entender y conseguir replicar el algoritmo FIRE, desarrollado por Fabrizio Russo en 1998.

Otros de los objetivos de este proyecto, consisten en realizar una serie de propuestas o modificaciones sobre la propuesta original. Estas propuestas radican en la optimización de los parámetros utilizados por los conjuntos difusos. En la propuesta original estos parámetros son fijos durante todo el proceso pero que en las propuestas que se realizarán, se modificarán para que sean variables con el fin de otorgar mayor libertad al algoritmo genético y tratar de conseguir mejores resultados.

Por último se pretende idear un sistema que sea capaz de entrenar con varias imágenes con ruido Sal y Pimienta para obtener un sistema final que pueda ser capaz de utilizarse sobre cualquier otra imagen con este ruido y conseguir unos resultados satisfactorios. Este sistema viene ideado porque creemos que no tiene sentido el idear un algoritmo para tratar de forma individual a una imagen. Pensamos que lo importante es idear un sistema que pueda ser aplicado a cualquier imagen.

2. CONCEPTOS PRELIMINARES

En este apartado se va a proceder a la definición de una serie de conceptos previos y necesarios para una facilitar la comprensión de las técnicas utilizadas.

2.1 Procesamiento digital de imágenes

El procesamiento digital de imágenes es el conjunto de técnicas que permiten mejorar la calidad o facilitar la búsqueda de información en una imagen por medio de una computadora.

El interés por el procesamiento digital de imágenes está fundamentado principalmente en dos áreas:

- Mejorar la calidad de la información para la interpretación humana.

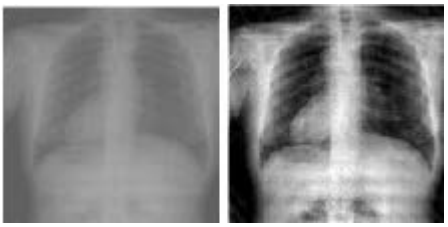


Figura 2-1: Ejemplo de mejora en la calidad de la imagen

- Procesar los datos de la imagen para su almacenamiento, transmisión y representación para la percepción autónoma de máquinas.

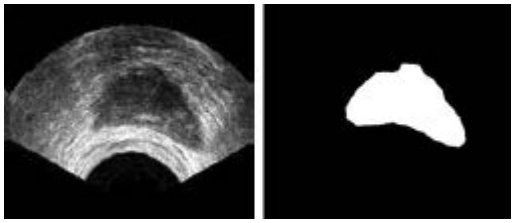


Figura 2-2: Ejemplo de procesado de imagen

Existe una gran cantidad de sectores donde el procesamiento digital de imágenes se utiliza de manera rutinaria. Una manera simple para categorizar las imágenes es dependiendo de su fuente física:

- **Radiación electromagnética (EM)**: Aplicaciones para medicina, astronomía, sistemas de radar, climatología, etc.
- **Acústica o Ultrasonidos**: Ultrasonido de bebé, tiroides, etc.
- **Electrónica**: Rayos de electrones en microscopios ópticos.

2.1.1 Imagen digital

Una imagen digital es una representación bidimensional de una imagen en forma de una matriz de F filas y C columnas. Si F y C son discretos y finitos entonces podemos decir que la imagen es digital.

En este proyecto se va a trabajar con imágenes representadas en escala de grises, es decir, cada píxel estará representado por un valor numérico que nos indica la intensidad del píxel, el cual estará entre el 0 y el 255.

Img(x,y) Є {0,.., 255} | con x en {1, …, F} e y en {1, …, C}

FxC = número totales de píxeles de la imagen digital.

Cuanto más bajo sea el valor, más oscuro será el píxel y viceversa. De esta forma, un píxel con valor 0 corresponde a negro y un píxel con valor 255 a blanco.

Ejemplo de representación de una imagen digital 3*3 en escala de grises:

$$Img = \begin{pmatrix} 0 & 4 & 103 \\ 167 & 255 & 175 \\ 47 & 200 & 60 \end{pmatrix}$$

Cuya imagen en la escala de grises sería:

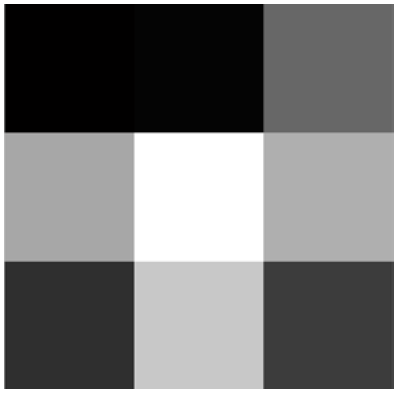


Figura 2-3: Ejemplo escala de grises de Img

En las imágenes digitales en color, cada píxel está representado por 3 valores numéricos denominados RGB (Red, Green, Blue) que son el nivel de intensidad en cada uno de esos 3 colores.

2.1.2 Ruido

El ruido digital es la variación de manera aleatoria de la intensidad o color en una imagen digital. Las principales fuentes de ruido surgen durante la adquisición y/o transmisión de las imágenes.

Durante la adquisición el rendimiento de los sensores de la imagen puede verse afectado por factores como las condiciones ambientales o la calidad del elemento de captura. Los niveles de luz y sensores de temperatura son diferentes para cada dispositivo de entrada y afectan de manera directa la cantidad de ruido resultante generado.

Durante la transmisión de imágenes también puede generarse ruido debido a una interferencia en el canal utilizado.

A la hora de modelar matemáticamente el ruido, se considera que en general, no depende ni del valor del píxel ni de su posición.

Algunos de los ruidos comúnmente encontrados son el gaussiano y sal y pimienta que se describen en las siguientes secciones.

2.1.2.1 Ruido Gaussiano

En el ruido de tipo Gaussiano, todos los píxeles que componen la imagen cambian su valor en base a una distribución normal o gaussiana. La distribución mostrada en la figura 2-4, el ruido está centrado en el valor Z (que se corresponde con el valor de intensidad del píxel) y tiene una desviación estándar de sigma.

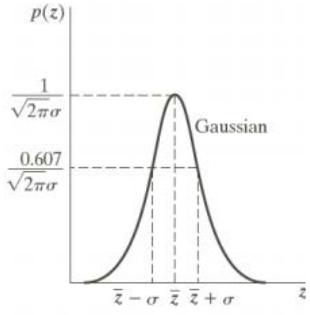


Figura 2-4: Función de distribución del ruido gaussiano

El efecto de este ruido es que en general se generan valores aleatorios que variarán poco respecto del valor original del píxel. La mayor parte de los valores establecidos quedaran similares al original pero con alguna pequeña modificación tal y como puede verse en la figura 2-5, donde la imagen de la izquierda muestra la imagen original y a su derecha la imagen con ruido gaussiano.

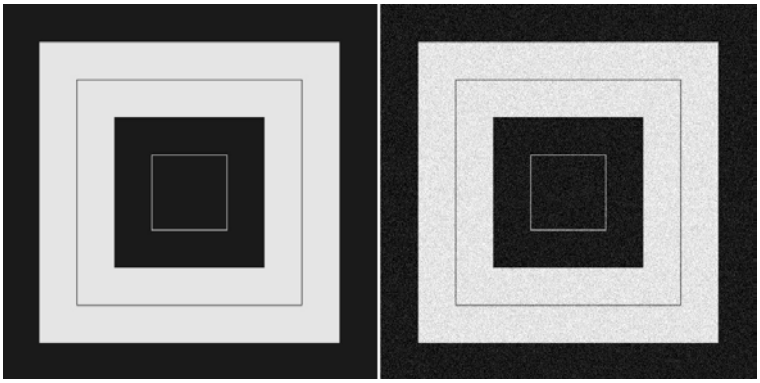


Figura 2-5: Ejemplo de ruido Gaussiano

2.1.2.2 Ruido Impulsivo (sal y pimienta)

El ruido Sal y pimienta es un caso particular del ruido impulsivo. El valor que toma el píxel no tiene relación con el valor original, sino que toma valores muy altos o muy bajos, es decir, casi blanco o negro. Matemáticamente este ruido se modela con una función de distribución no gaussiana o escalón como la mostrada en la figura 2-6.

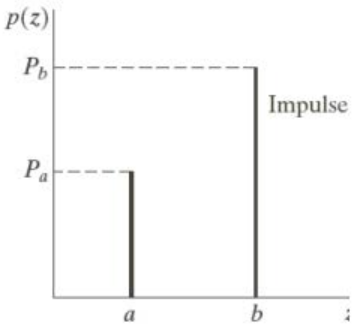


Figura 2-6: Función de distribución del ruido Sal y Pimienta

El efecto final de este ruido es que ciertos píxeles de forma aleatoria cambian a el valor máximo (sal=blanco) o el mínimo (pimienta=negro), tal y como puede verse en la figura 2-7, donde la imagen de la izquierda muestra la imagen original y a su derecha la imagen con ruido impulsivo Sal y Pimienta.

Figura 2-7: Ejemplo de ruido Sal y Pimienta

2.1.3 Filtrado

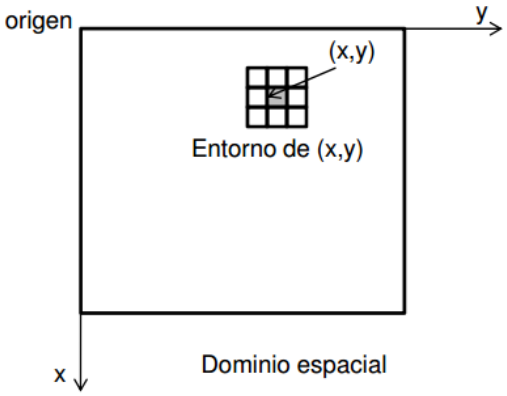
Los filtros constituyen uno de los principales modos de operar en el procesamiento de imágenes digitales. Los principales objetivos que se persiguen con la aplicación de filtros son:

- **Suavizar la imagen**: reducir la cantidad de variaciones de intensidad entre píxeles vecinos.
- **Eliminar ruido**: tratar aquellos píxeles cuyo nivel de intensidad es muy distinto al de sus vecinos.
- **Realzar bordes:** destacar los bordes de los objetos contenidos en una imagen.
- **Detectar bordes**: detectar los píxeles donde se produce un cambio brusco de intensidad.

Los filtros se pueden clasificar en dos grandes grupos: filtros que trabajan en el dominio del espacio y filtros en el dominio de la frecuencia.

- **Filtros en el dominio de la frecuencia**: las operaciones se realizan sobre la transformada de Fourier de la imagen.

- **Filtros en el dominio del espacio**: operan directamente sobre los píxeles de la imagen. Las operaciones espaciales de filtrado se definen en un entorno de vecindad sobre el píxel a transformar localizados en la posición (x,y). Generalmente, se entenderá como el entorno de vecindad del píxel localizado en (x,y), como una ventana de MxN píxeles centrada en dicho píxel.



Los filtros en el dominio del espacio pueden clasificarse en filtros lineales y no lineales.

  - **Filtros lineales**: filtros que operan sobre la vecindad mediante operaciones lineales y que se basan en máscaras de convolución. Algunos ejemplos de los más conocidos son los basados en diferentes tipos de medias. Estos filtros se caracterizan por coger el valor del píxel central localizado en (x,y), operar teniendo en cuenta su ventana de vecindad y sustituyendo el valor original del píxel por el obtenido con la media correspondiente.

- **Filtro de la media**:

$$g(x,y) = \sum_{s=-a}^{a} \sum_{t=-b}^{b} \frac{1}{mxn} f(x+s, y+t)$$

- **Filtro de la media geométrica**:

$$Mg = \prod_{(x,y)\in W} [f(x,y)]^{1/nm}$$

- **Filtro de la media armónica**:

$$Mar = \frac{nm}{\sum_{(x,y)\in W} \frac{1}{f(x,y)}}$$

- **Otros**: filtro de la media contra-armónica, gaussianos, etc.

- **Filtros no lineales**: también operan sobre la vecindad pero no mediante sumas/multiplicaciones, sino basados en proyecciones ordenadas. Algunos de los más conocidos son:
  - **Filtro de la mediana**: se reemplaza cada píxel por la mediana de los píxeles del vecindario.
  - **Filtro del mínimo/máximo**: se reemplaza cada píxel por el valor máximo o mínimo del vecindario
  - **Otros**: Alpha-Media, punto medio, filtro adaptativo de reducción local, etc.

2.1.4 Criterios de fidelidad

Cuando se tratan imágenes para la eliminación de ruido, es necesario saber la calidad del tratamiento aplicado. Para ello se cuantifica el error (disimilitud) entre la imagen original y la imagen fitrada. Así pues, se aplica la siguiente operación matemática:

$$e(x,y) = \mathrm{f'(x,y)} - \mathrm{f(x,y)}$$

donde f(x,y) es el valor del píxel en la imagen original y f'(x,y) es el valor del píxel en la imagen filtrada.

Algunos de los más comunes para medir la calidad de las imágenes filtradas son:

- **Error cuadrático medio (Mean Square Error)**

$$MSE = \frac{1}{MN}\sum_{x=1}^{M}\sum_{y=1}^{N}(f'(x,y) - f(x,y))^2$$

Cuanto menor sea el valor de MSE, más parecida es la imagen filtrada a la imagen original. En la situación ideal en la que el filtrado es perfecto, el MSE obtenido será 0.

- **Peak Signal to Noise Ratio (PSNR)**

$$PSNR = 10\log_{10}\frac{(L-1)^2}{MSE}$$

Cuanto mayor sea el valor de PSNR, más parecida es la imagen filtrada a la imagen original. En la situación ideal en la que el filtrado es perfecto, el PSNR obtenido será $+\infty$.

2.2 Lógica difusa

En 1965 Lotfi Zadeh introdujo la teoría de conjuntos difusos.

Un conjunto difuso A definido sobre un universo finito y no vacío U = $\{u_1,\ldots,u_n\}$ viene dado por:

A: $\{(u_i, \mu_A(u_i)) \mid u_i \in U\}$

donde μ_A: U→ [0,1] es tal que $\mu_A(u_i) \in [0,1]$ denota el grado de pertenencia del elemento u_i al conjunto A.

Por lo tanto, cuanto más cerca esté el valor de $\mu_A(u_i)$ a 1, mayor será el grado de pertenencia de u_i en A. Cuando A es un conjunto en el sentido clásico, su función de pertenencia puede tomar sólo dos valores 0 y 1, es decir $\mu_A(u_i)$ = {1 o 0} cuando u_i pertenece o no pertenecen a A, respectivamente.

En el mundo real existe mucho conocimiento no-perfecto, es decir, conocimiento vago, impreciso, incierto, ambiguo, inexacto, o probabilístico por naturaleza. El razonamiento y pensamiento humano frecuentemente conlleva información de este tipo, probablemente originada por la inexactitud inherente a los conceptos humanos y del razonamiento basado en experiencias similares pero no idénticas a experiencias anteriores. La lógica difusa, es una lógica alternativa a la lógica clásica que pretende introducir un grado de vaguedad en los conceptos que califica para modelar lo descrito anteriormente.

El problema principal surge de la poca capacidad de expresión de la lógica clásica. Supongamos por ejemplo que tenemos un conjunto de personas que intentamos agrupar según su altura, clasificándolas en altas o bajas. La solución que presenta la lógica clásica es definir un umbral de pertenencia (por ejemplo, un valor que todo el

mundo considera que de ser alcanzado o superado, la persona en cuestión puede llamarse alta). Si dicho umbral es 1.80, todas las personas que midan 1.80 o más serán altas, mientras que las otras serán bajas. Según esta manera de pensar, alguien que mida 1.79 será tratado igual que otro que mida 1.50, ya que ambas han merecido el calificativo de bajas. Sin embargo, si dispusiéramos de una herramienta para caracterizar las alturas de forma que las transiciones fueran suaves, estaríamos reproduciendo la realidad mucho más fielmente.

Asimismo, no hay un valor cuantitativo que defina el término joven. Para algunas personas, 25 años es joven, mientras que para otras, 35 es joven. Incluso el concepto puede ser relativo al contexto. Un presidente de gobierno de 35 años es joven, mientras que un futbolista no lo es. Sin embargo hay cosas que están claras: una persona de 1 año es joven, mientras que una de 100 años no lo es. Pero una persona de 35 años tiene posibilidades de ser joven (normalmente dependen del contexto). Para representar este hecho, definiremos el conjunto joven de modo que cada uno de sus elementos (edades) pertenezca a él con cierto grado (posibilidad).

Por ejemplo podemos modelar el conjunto difuso joven para la edad utilizando una función de pertenencia como la mostrada en la figura 2-8. En este caso, una persona será completamente joven cuando tenga menos de 25 años y no será para nada joven cuando tenga más de 50. En las edades intermedias, los grados de pertenencia irán siendo menores progresivamente.

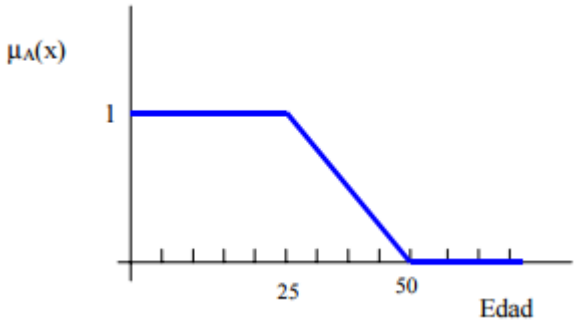


Figura 2-8: Función de pertenencia de joven si el U es continuo

La lógica difusa es útil y efectiva en las siguientes situaciones:

- En procesos complejos, si no existe un modelo de solución sencillo.
- Cuando haya que introducir la experiencia de un operador “experto” que se base en conceptos imprecisos.
- Cuando ciertas partes del sistema a controlar son desconocidas y no pueden medirse de forma fiable.
- Cuando el ajuste de una variable puede producir el desajuste de otras.

Algunos de los términos más utilizados en el ámbito de los conjuntos difusos son:

- El núcleo de un subconjunto difuso A es el conjunto de los elementos u_i que pertenecen totalmente a A, es decir que verifican $\mu A(u_i) = 1$
- El soporte de subconjunto difuso A es el conjunto de los elementos u_i que pertenecen, en cierto grado, a A. Es decir que verifican $\mu A(u_i) > 0$.
- Variable difusa es una variable en la que se utilizan conjuntos difusos para representar sus estados mediante conceptos lingüísticos.
- Dos conjuntos difusos A y B son iguales si y solo si sus funciones características $\mu A(u_i)$ y $\mu B(u_i)$ son iguales.

Las operaciones básicas entre conjuntos difusos son las siguientes:

- El conjunto complementario $\neg A$ de un conjunto difuso A es aquel cuya función de pertenencia viene definida por $\mu A(u_i) = 1 - \mu A(u_i)$
- La unión de dos conjuntos difusos A y B es un conjunto difuso $A \cup B$ en U cuya función de pertenencia es $\mu_{A \cup B}(u_i) = S[\mu A(u_i), \mu B(u_i)]$
- La intersección de dos conjuntos difusos A y B es un conjunto difuso $A \cap B$ en U con función característica $\mu_{A \cap B}(u_i) = T[\mu A(u_i), \mu B(u_i)]$

S y T son t-normas y t-conormas respectivamente y se explican en las secciones siguientes.

Estas tres operaciones definidas para conjuntos difusos cumplen, al igual que en la teoría clásica de conjuntos, asociatividad, conmutatividad y distributividad, así como las leyes de Morgan. Sin embargo, existen dos leyes fundamentales de la teoría clásica de conjuntos como son el Principio de contradicción $A \cup \neg A$ = U, y el Principio de exclusión: $A \cap \neg A = \emptyset$ que no se cumplen en la teoría de conjuntos difusos.

2.2.1 Operador t-norma

Una función T: $[0,1]^2 \rightarrow [0,1]$ es una t-norma si satisface las siguiente propiedades:

- Condición de contorno T(x,1)= x para todo x ϵ [0,1];
- Monótona, T(x,y) ≤ T(z,u) si x ≤ z e y≤u;
- Conmutativa, T(x,y) = T(y,x) para todo x,y ϵ [0,1];
- Asociativa, T(T(x,y),z) = T(x,T(y,z)) para todo x,y,z ϵ [0,1];

Ejemplos de t-norma:

- Mínimo: T(x,y) = min{x,y}
- Producto algebraico: T(x,y)= x*y
- Lukasiewicz: T(x,y) = max{0, x + y -1}

2.2.2 Operador t-conorma

Una función S: $[0,1]^2 \rightarrow [0,1]$ es una t-conorma si satisface las siguiente propiedades:

- Condición de contorno S(x,0)= x para todo x ϵ [0,1];
- Monótona, S(x,y) ≤ S(z,u) si x ≤ z e y≤u;
- Conmutativa, S(x,y) = S(y,x) para todo x,y ϵ [0,1];
- Asociativa, S(S(x,y),z) = S(x,S(y,z)) para todo x,y,z ϵ [0,1];

Ejemplos de t-conorma:

- Máximo: S(x,y) = max{x,y}
- Suma probabilística S(x,y)= x+y-x*y
- Lukasiewicz: S(x,y) = min{1, x + y}

2.3 Algoritmos genéticos

Los algoritmos genéticos son métodos adaptativos que pueden usarse para resolver problemas de búsqueda y optimización. Están basados en el proceso evolutivo de los organismos vivos. A lo largo de las generaciones, las poblaciones evolucionan en la naturaleza de acuerdo con los principios de la selección natural y la supervivencia de los más fuertes, postulados por Darwin (1859). Por imitación de este proceso, los algoritmos genéticos son capaces de ir creando soluciones para problemas del mundo real. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas.

Algunas de las ventajas de estos algoritmos son que hacen un uso inteligente de las búsquedas llevadas a cabo en problemas de optimización y aunque son aleatorios, se ayudan de información pasada para orientar la búsqueda en el futuro. Además son más robustos que otros métodos de la inteligencia artificial. Es decir, son menos sensibles a pequeñas variaciones en los parámetros iniciales o a perturbaciones debidas a ruido, información incompleta, etc. Igualmente la búsqueda en espacios de soluciones muy grandes es más efectiva que con otros métodos de Inteligencia Artificial. En particular, son muy apropiados para llevar a cabo búsquedas en espacios de soluciones muy grandes o para lograr combinaciones óptimas de elementos que, de otra forma, resultaría temporalmente muy costoso, si no imposibles de obtener.

Los algoritmos genéticos parten de una serie de individuos o soluciones que constituyen la población. Éstos se utilizan para generar los individuos de la siguiente población. Las soluciones para crear una nueva población se eligen teniendo en cuenta su adaptación al problema a resolver. Este proceso se repite hasta que se cumpla una condición dada (número de iteraciones máximas alcanzadas, solución perfecta encontrada, etc.)

Algunos de los conceptos básicos de los diferentes elementos y componentes que forman un algoritmo genético son:

- **Cromosoma o individuo**: Conjunto de genes. Un cromosoma o individuo representa la solución a un problema dado en forma de genes, que son los que representan los parámetros del sistema a optimizar o del problema a resolver.
- **Población**: Conjunto de individuos que utiliza el algoritmo genético para solucionar el problema.
- **Fitness (Adaptación, calidad)**: Valor que se asigna a cada individuo en función de lo lejos o cerca que se haya de la solución. Cuanto mayor es el valor de fitness de un cromosoma, mejor solución representa.
- **Función de fitness**: Función que asigna a cada individuo su fitness. Depende de cada problema específico.
- **Cruce**: Operación que toma dos individuos y combina sus cromosomas para obtener nuevos individuos.
- **Mutación**: Operación que modifica ligeramente un gen en un individuo.
- **Selección**: Operación que escoge individuos realizar la operación de cruce.

El esquema básico de un algoritmo genético estándar sigue estos pasos

COMIENZO

INICIALIZAR la población aleatoriamente

POR CADA INDIVIDUO evaluar su calidad utilizando la función de fitness

REPETIR HASTA condición HACER

    1-SELECCIONAR padres

    2-CRUZAR los padres

    3-MUTAR los descendientes resultantes

    4-POR CADA DESCENDIENTE evaluar su calidad utilizando la función de fitness

    5-SELECCIONAR los individuos para formar la siguiente generación

FIN

Como se ha descrito anteriormente, son necesarios una serie de conceptos o términos para una correcta síntesis de este tipo de algoritmos.

Algunos de los componentes más importantes anteriormente explicados poseen una serie de variantes o posibilidades dependiendo del tipo del problema, codificación del cromosoma, cruce, etc. Ciertos componentes requieren una mención y explicación más precisa.

- **Representación**: Es necesario relacionar el problema en el mundo real con nuestras posibilidades de codificación. Hay dos codificaciones muy habituales que son la binaria y la real. En la binaria, cada cromosoma es una secuencia de ceros y unos:

  Cromosoma 1: 001001100110

  Cromosoma 2: 101111000010

  Existen otros tipos de representaciones como las basadas en permutaciones, codificación en árbol, etc. La elección de cual escoger dependerá del tipo de problema.

- **Población**: La población representa todas las soluciones que existen en un momento dado. La diversidad de una población se refiere a la variedad de soluciones entre los individuos que componen la población.

- **Mecanismo de selección de padres**: Este mecanismo, que debe también fijarse de antemano, escoge entre la población los individuos mejor adaptados para generar la siguiente población. Un individuo es un padre o progenitor si ha sido seleccionado para generar nuevas soluciones (individuos). En general, los individuos de mejor calidad tienen más posibilidades de ser escogidos como padres. Sin embargo, los individuos de baja calidad tienen una probabilidad positiva aunque pequeña de ser elegidos, para evitar caer en óptimos locales.

  Algunos ejemplos de estos mecanismos de selección son:

  - **Método de la Ruleta**: La probabilidad de que un individuo sea seleccionado es proporcional a su fitness.

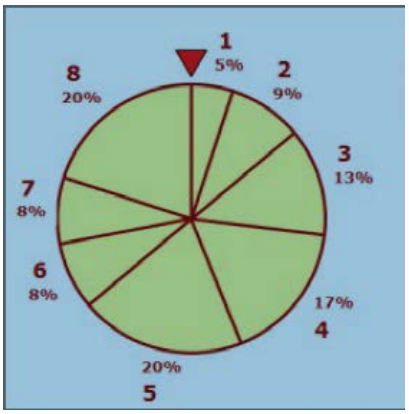


Figura 2-9: Método de la ruleta

En este método se puede visualizar intuitivamente como una ruleta, tal y como se muestra en la figura 2-9 en la que se representa una población de 8 individuos. El porcentaje representado es proporcional al valor de fitness de cada uno de ellos. En particular, como el cromosoma 5 presenta un valor mayor, tendrá más opciones de ser elegido para cruzarse más veces.

- **Método del Torneo:** En este método no se necesita un conocimiento global de lo óptima que es una solución ni de todas las soluciones simultáneamente. El método se basa en comparar k cromosomas cada vez. La probabilidad de que un individuo sea seleccionado depende de varios factores.
  1. El tamaño k del torneo. A mayor número de individuos, más probabilidad de ser seleccionado.
  2. La posibilidad de que el mejor adaptado del torneo sea elegido. En general, esta probabilidad es 1, es decir, se fuerza.
  3. Si los individuos son seleccionados con o sin sustitución. En el primer caso, incluso el peor adaptado podría ser seleccionado.

- **Mutación**: Un operador de mutación actúa sobre un único individuo y lo devuelve ligeramente modificado. El papel de las mutaciones en los algoritmos genéticos es proporcionar diversidad. Sin embargo, existen algoritmos evolutivos en los que no se usan y otros en los que son la única fuente de cambio utilizada para alcanzar la solución. La mutación se lleva a cabo por medio de operadores estocásticos (probabilísticos aleatorios) y permiten en ciertos casos garantizar que los algoritmos genéticos alcanzan una solución.

  En el ejemplo de una codificación binaria, Cada gen puede cambiar su valor (de 0 a 1 o de 1 a 0) con probabilidad p, como se puede ver en la figura 2-10.

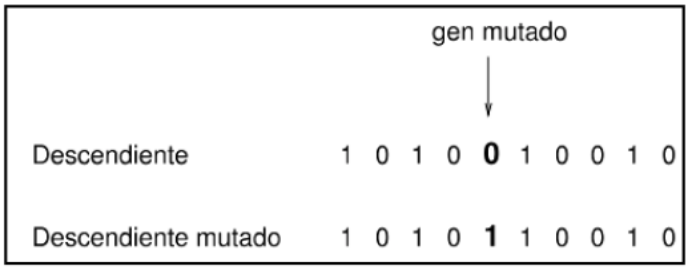


Figura 2-10: Ejemplo de mutación de un cromosoma

  La probabilidad de mutación para algoritmos genéticos se toma muy baja (del orden de 0,05 o inferior). Existen muchos métodos de mutación dependiendo del tipo de representación (mediante enteros, en punto flotante, mediante permutaciones).

- **Cruce**: Como anteriormente se ha mencionado, es un operador que toma dos individuos, los combina y devuelve uno o dos descendientes. En los algoritmos

genéticos es el principal operador de búsqueda de soluciones. Existen muchos métodos de cruce dependiendo del tipo de codificación del cromosoma que se tenga en cada problema. Para codificaciones binarias cabe destacar:

- **Cruce en un punto**: Dados dos cromosomas de longitud N, se elige aleatoriamente un entero l entre 0 y N-1 y se toman los l primeros genes del cromosoma 1 con los N-l segundos genes del cromosoma 2 y l primeros genes del cromosoma 2 con los N-l segundos genes del cromosoma 1 tal y como se muestra en la figura 2-11.

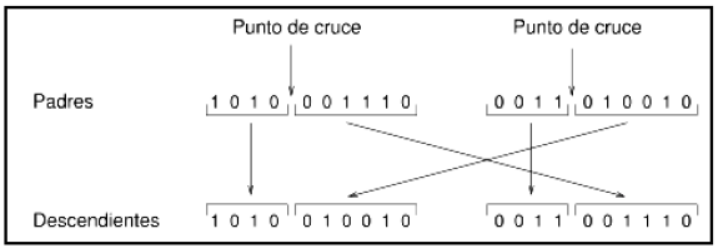


Figura 2-11: Cruce sobre un punto

- **Cruce de n puntos:** Es igual al anterior pero con n puntos de cruce.
- **Otros**: Cruce uniforme, etc.

Para codificaciones no binarias cabe destacar:

- **Cruce por recombinación aritmética**: A partir de dos cromosomas se hace la media para todas las posiciones.

- **Cruce PCBLX**: Dados dos cromosomas C1 = ($c1_1$,…, $c1_n$) y C2 = ($c2_1$,…, $c2_n$), donde $c1_i$, $c2_i \in [a_i,b_i]$. Se generan dos descendientes, Hk = (hk_1,…, hk_i,…, hk_n), k =1,2 donde hk_i se genera aleatoriamente en el intervalo $[l_i, u_i]$.

K=1	K=2
$l_i = \max \{a_i, c1_i - D\}$ $u_i = \min \{b_i, c1_i + D\}$ $D = \lvert c1_i - c2_i \rvert$	$l_i = \max \{a_i, c2_i - D\}$ $u_i = \min \{b_i, c2_i + D\}$ $D = \lvert c1_i - c2_i \rvert$

- **Otros**: Recombinación aritmética total (en punto real), parcialmente mapeado (por permutaciones), cruce de aristas, cruzamiento basado en orden, cíclico, etc.

- **Método de reemplazo de la población**: Consiste en escoger los individuos que formarán parte de la siguiente población. En general es determinista: se eliminan los individuos más viejos o peor adaptados de la generación anterior.
  - **Sustitución basada en la edad**: Cada individuo solo está presente en el algoritmo durante un número fijado de generaciones. Por supuesto, esto puede llevar a la eliminación de los mejor adaptados lo cual puede ser un problema.
  - **Reemplazar los peores individuos**: Se eliminan los peores individuos. Puede producir convergencia prematura.
  - **Elitismo**: Se mantiene el mejor individuo en la siguiente población y el resto de individuos se sustituyen por los hijos. El mayor elitismo se consigue al combinar los padres con los hijos y eligiendo los mejores.

2.4 Algoritmo FIRE

Los operadores FIRE (Fuzzy Inference Ruled by Else-action) son una clase de operadores no lineales para el procesamiento de imágenes mediante el uso de reglas difusas. Este algoritmo propone el uso de un algoritmo genético mediante el cual se describe un método para la generación automática de la base de reglas difusas. El creador de los operadores FIRE es Fabrizio Russo (Universidad de Trieste, Italia, en el año 1998).

El algoritmo FIRE está enfocado en la eliminación del ruido impulsivo Sal y Pimienta y preservación del detalle. También es utilizado como método de detección de bordes pero no está contemplado en este proyecto.

Una vez esta pequeña introducción se va a proceder a explicar cada proceso y componentes de este tipo de filtrado.

El algoritmo FIRE se basa en operaciones mediante las cuales se va tratando cada píxel de la imagen teniendo en cuenta su ventana de vecindad (3x3).

X_1	X_2	X_3
X_8	X	X_4
X_7	X_6	X_5

Figura 2-12: Ventana de vecindad para cada píxel

En el proceso del tratamiento de la imagen con ruido, se obtienen las llamadas variables de entrada Δ para cada píxel en función de su vecindad de la siguiente manera:

$$\Delta x_j(n) = x_j(n) - x(n)$$

Es decir, cada variable de entrada Δx_j se calculará restando el valor de cada vecino menos el píxel central con lo que tenemos 8 variables de entrada para cada regla. De

esta forma las reglas tendrán 8 antecedentes y vendrán representadas por cadenas de unos y ceros como se puede ver en la figura 2-12, donde las variables 2, 4 y 6 conformarían el antecedente de la regla.

0	1	0	1	0	1	0	0

Figura 2-13: Ejemplo de representación de una regla

La regla representada en la Figura 2-13 se establece de la siguiente manera:

$$IF\ (\Delta x_2, LP)\ AND\ (\Delta x_4, LP)\ AND\ (\Delta x_6, LP)\quad THEN\ (\Delta y, \mathrm{PO})$$

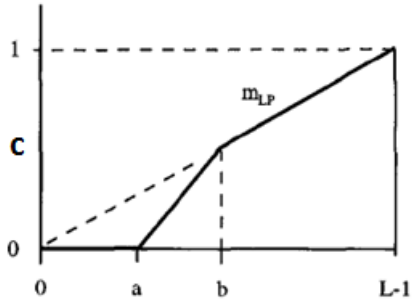
$$IF\ (\Delta x_2, LN)\ AND\ (\Delta x_4, LN)\ AND\ (\Delta x_6, LN)\quad THEN\ (\Delta y, \mathrm{NE})$$

Ilustración 2-1: Representación de una regla LP y LN

Donde PO (positivo) y NE (negativo) representan singletons centrados en L-1 y –L+1.

Al aplicar cada regla como PO y NE, hace que estén diseñadas para hacer frente a los impulsos de ruido positivo y negativo (Sal y Pimienta).

Una vez calculadas estas variables de entrada, se fuzzifican mediante dos conjuntos difusos etiquetados LP (large positive) y LN (large negative). La función de pertenencia del conjunto difuso LP es la siguiente:



$$m_{LP}(x) = \begin{cases} 0 & si\ \ 0 < \ x < a \\ \frac{x-a}{b-a} * c & si\ a \le x \le b \\ c + \frac{x-b}{L-1-b} * (1-c) & si\ \ b \le x \le L-1 \end{cases}$$

Figura 2-14: Función de pertenencia del conjunto LP

Para fuzzificar las variables de entrada mediante el conjunto difuso LN simplemente se realiza:

$$m_{LN}(x) = m_{LP}(-x)$$

Es decir, es el antónimo de LP. Para calcularlo se cambian de signo las variables de entrada y se realiza el proceso exactamente igual que en LP.

Para poder realizar este proceso (fuzzificar las variables de entrada mediante LP y LN) es necesario un cálculo previo del parámetro 'a'.

El cálculo del parámetro 'a' viene formulado a partir de una nueva función de pertenencia:

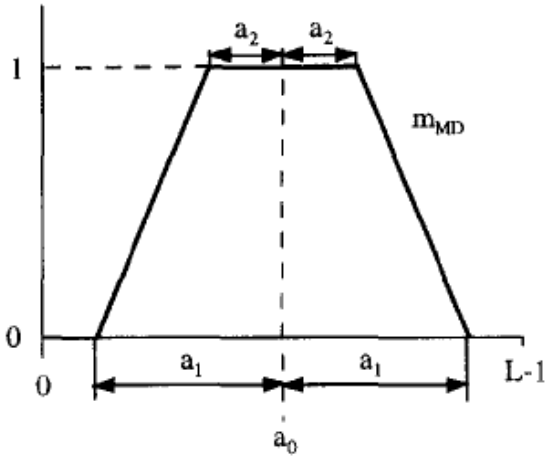


Figura 2-15: Función de pertenencia para calcular la variable "a"

donde $a = a_{max} * m_{MD}(x)$, y $a_{max} = b$.

Tanto los parámetros a_0, a_1, a_2 de esta función de pertenencia, como los parámetros b y c de la anterior, vienen fijados por el autor y sus valores son 128, 120, 40, 127 y 0.5 respectivamente. Estos valores no cambian en ningún momento durante el proceso del algoritmo FIRE. L es el máximo del rango de los valores que puede tomar un píxel, es decir, 255.

Después de realizar el proceso de fuzzificacion (LP y LN), estos valores fuzzificados serán los que realmente les lleguen a las variables de las reglas como se muestra en la Ilustración 2-1.

En general se debe considerar más de un patrón a la hora de aplicar una regla con el fin de tener en cuenta las posibles combinaciones de ruido en píxeles adyacentes. Como se ha comentado anteriormente, está generación de reglas automáticas se realizara en el proceso del algoritmo genético.

El valor de salida Δy se calcula de la siguiente manera:

$$\Delta y = (L - 1)(\lambda_1 - \lambda_2)$$

donde

$$\lambda_1 = MAX\{MIN\{m_{LP}(\Delta x_j), j \in A_i\}, i = 1, \dots, N\}$$

$$\lambda_2 = MAX\{MIN\{m_{LN}(\Delta x_j), j \in A_i\}, i = 1, \dots, N\}$$

Una vez calculado el valor de salida Δy, nuestro nuevo valor para el píxel central será igual a la suma de su valor y Δy. Si no se satisface ninguna regla (todos los bits son iguales a 0), el valor del píxel central no se modifica (se quedará con el mismo valor).

Por último, el algoritmo FIRE se apoya en el uso de un algoritmo genético para generar la base de reglas de forma automática.

La generación de estas reglas se realiza de la siguiente manera:

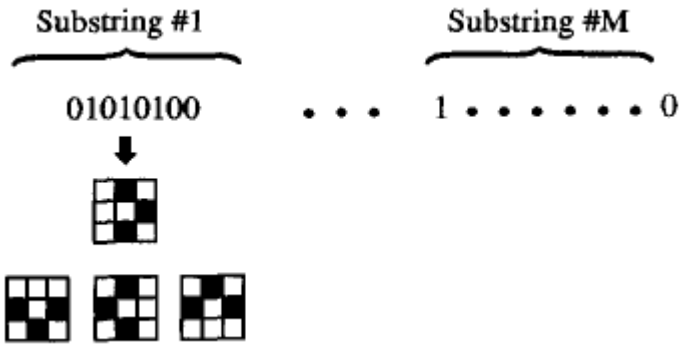


Figura 2-16: Ejemplo de generación de reglas

Cada cromosoma es una cadena binaria de M x 8 bits que representa una base de reglas completa. Cada una de estas cadenas contiene M substrings. Cada substring representa un total de 8 reglas debido al siguiente motivo:

Por cada substring realizaremos tres giros (90, 180 y 270 grados). Para el ejemplo anterior, mostrado en la Figura 2-16, las cadenas resultantes son las siguientes:

0	1	0	1	0	1	0	0	Inicial
0	0	0	1	0	1	0	1	90 grados
0	1	0	0	0	1	0	1	180 grados
0	1	0	1	0	0	0	1	270 grados

Con este procedimiento generaremos 3 reglas más. Si las añadimos a la regla inicial son 4 reglas. Estas cuatro reglas se aplicaran tanto a LP y a LN por lo que se pueden considerar 8 reglas. Para el resto de substrings se realiza el mismo proceso por lo que al final tenemos un total de M * 4 reglas * 2(LP y LN) para aplicar.

El algoritmo empieza con una población de individuos generada aleatoriamente. Después se aplica todo el método explicado anteriormente para tratar cada píxel de la imagen con ruido y obtener una imagen filtrada.

Una vez recorrida la imagen y con el fin de poder medir el error que hemos generado con la base de reglas que representa el cromosoma tratado en cada momento, la función de Fitness (será nuestra medida para evaluar a cada individuo de la población) viene definida como:

$$F = \frac{1}{MSE} = \frac{(N_1 - 2)(N_2 - 2)}{\sum_{n_1=1}^{N_1-2}\sum_{n2=1}^{N_2-2}(y(n_1, n_2) - s(n_1, n_2))^2}$$

donde “y” y “s” se corresponden con la imagen limpiada y la imagen original sin ruido. La única diferencia se encuentra en que al tratar siempre con ventanas 3x3, al calcular el error no se tienen en cuenta los píxeles de la primera y última fila y columnas porque no se tratan en ningún momento.

Este proceso se realiza para cada individuo de la población y sucesivas generaciones. Cuando el algoritmo concluya se habrá obtenido la mejor base de reglas para esa imagen.

3. PROPUESTAS

En este capítulo se presentan las propuestas planteadas en el proyecto en las que se utilizan las técnicas descritas en las secciones anteriores. Todas ellas mantienen la esencia de la propuesta del algoritmo original FIRE pero incorporando y/o modificando ciertos matices con el fin de buscar una mejora en el desarrollo de la misma.

3.1 Implementación del algoritmo FIRE

En este apartado se van a describir ciertos aspectos técnicos de la configuración en la implementación del algoritmo FIRE. El proceso y su mecánica siguen exactamente los mismos pasos que los descritos en el apartado Algoritmo FIRE, salvo por una ampliación explicada más adelante.

La representación de un cromosoma viene definido como una cadena de Mx8 bits.

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

Este cromosoma ha sido fijado como el inicial. Tiene 24 elementos por lo que tiene 3 substrings de 8 bits y como se ha descrito en la generación de reglas, se generaran 24 reglas.

Los parámetros de los conjuntos difusos a fuzzificar son fijos durante todo el programa son:

a_0	128
a_1	120
a_2	40
b	127
c	0,5

En el proceso del ajuste del algoritmo genético, el autor no detalla nada acerca de ninguno de sus parámetros. No impone método de selección de progenitores, cruce, mutación, etc. Por ello, se va a detallar los utilizados en ésta y en las demás propuestas. Todos ellos están explicados detalladamente en los Conceptos preliminares.

Al iniciar el algoritmo genético, la población inicial está formada por 20 individuos (el cromosoma inicial impuesto y 19 aleatorios).

La condición de parada viene determinada por dos parámetros:

- Se llegue a un numero de iteraciones máximas = 200
- Durante 10 generaciones no mejore el Fitness del mejor cromosoma en 0,000001. Se recuerda que el $Fitness = \frac{1}{MSE}$

La selección de progenitores se realiza mediante el método de la ruleta.

Para cruzar los progenitores se usa el cruce por un punto. En nuestro caso, al estar compuesto el cromosoma por M cadenas de 8 bits, se realiza un cruce por cada Substring (en nuestro caso 3). Además, el cruce se realiza con una probabilidad de 0,9. En caso de no producirse, los descendientes serán los propios progenitores.

La mutación se realiza sobre cada gen y tiene una probabilidad de realizarse de 0,05.

Una vez se evalúan los nuevos descendientes (20 individuos), la siguiente generación estará formada por los 20 mejores individuos (de entre la generación actual y los descendientes nuevos). Es decir, se aplica el mayor nivel de elitismo posible para guiar la búsqueda completamente por las mejores soluciones.

Una vez ha concluido el algoritmo genético se habrá obtenido el cromosoma que representa la mejor base de reglas para una imagen inicial con ruido.

Después de obtener el cromosoma ganador se limpia la imagen con ruido inicial y a continuación, se vuelve a aplicar la misma base de reglas utilizando la imagen filtrada anteriormente. Esta segunda imagen será la solución final del algoritmo.

A esta extensión se le ha llamado proceso de recuperación de la mejor imagen y limpieza.

3.2 Ajuste de los parámetros del conjunto difuso LP

Esta propuesta presenta dos diferencias respecto a la anterior.

El algoritmo original deja fijos los parámetros $a_0, a_1, a_2, b\ y\ c$, por lo que puede estar limitando la calidad de las soluciones. Por este motivo se propone añadir estas variables dentro del cromosoma.

La estructura de un cromosoma para esta versión queda definida de la siguiente manera:

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5

El propósito de la modificación de los parámetros de los conjuntos difusos es para dar una mayor libertad al algoritmo genético para obtener mejores resultados.

A la hora de realizar el cruce, se efectuará en dos partes. Una primera (la parte binaria) de la misma manera que en la propuesta anterior y la segunda (parte real) se realizará mediante el cruce PCBLX, también explicado en los conceptos preliminares.

Un detalle a tener en cuenta tanto a la hora de crear estos nuevos parámetros reales de forma aleatoria como en el cruce es que siempre tiene que cumplirse $a_1 \geq a_2$ por la propia definición del conjunto difuso para obtener el valor "a".

Otra puntualización que merece ser destacada es que los parámetros $a_0, a_1, a_2\ y\ b$ se tienen que encontrar siempre en el rango [0,255] y c en [0,1].

El resto del procedimiento (aplicación del algoritmo FIRE y proceso de recuperación y limpia) es idéntico al explicado en la propuesta anterior.

3.3 Ajuste de los parámetros del conjunto difuso LP y LN

En la propuesta anterior se han añadido los parámetros $a_0, a_1, a_2, b\ y\ c$ en el cromosoma para que actúen de manera variable por cada cromosoma. En esta propuesta se realizará una distinción entre parámetros $a_0, a_1, a_2, b\ y\ c$ para LP y $a_0, a_1, a_2, b\ y\ c$ para LN.

Por lo tanto la estructura de un cromosoma en esta versión quedará definida de la siguiente manera:

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5	127	120	40	127	0,5

Es decir, se introducen 5 parámetros para el conjunto difuso LP y otros 5 para LN. De esta forma dichos conjuntos difusos ya no son antónimos entre ellos. A la hora de realizar los procesos de fuzzificación LP y LN, cada conjunto difuso tendrá sus propios parámetros en vez de utilizar los dos lo mismos como en la versión anterior.

Tanto el procedimiento (aplicación del algoritmo FIRE y proceso de recuperación y limpia) como la configuración del algoritmo genético son exactamente igual al explicado en la propuesta Implementación del algoritmo FIRE.

3.4 Optimización de los parámetros del conjunto difuso LP para cada variable de entrada

En este apartado se quiere dotar de aún más libertad de aprendizaje al algoritmo genético. Volviendo a la estructura tomada en la propuesta Ajuste de los parámetros del conjunto difuso LP (añadir los parámetros $a_0, a_1, a_2, b\ y\ c$ al cromosoma) se va a realizar una optimización sobre este planteamiento. Se debe destacar que en esta propuesta LN siempre será el antónimo del correspondiente LP.

Se añadirán una cadena de parámetros $a_0, a_1, a_2, b\ y\ c$ por cada variable de entrada Δx_j. Por lo tanto, al haber 8 variables de entrada nuestro nuevo cromosoma tendrá 8*5 parámetros (parte real) y la parte binaria (igual que siempre).

La representación de la estructura de un cromosoma para esta nueva propuesta es:

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127

0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5

En esta propuesta al igual que en la propuesta Ajuste de los parámetros del conjunto difuso LP, no se hará distinción de parámetros para los conjuntos difusos LP y LN. La única diferencia es que el proceso de fuzzificar los conjuntos difusos LP y LN habrá uno por cada componente Δx_j $(LP_1, \dots, LP_8)$.

Al igual que en todas las propuestas, el procedimiento (aplicación del algoritmo FIRE y proceso de recuperación y limpia) como la configuración del algoritmo genético son exactamente igual al explicado en la propuesta Implementación del algoritmo FIRE.

3.5 Algoritmo general de limpieza con FIRE

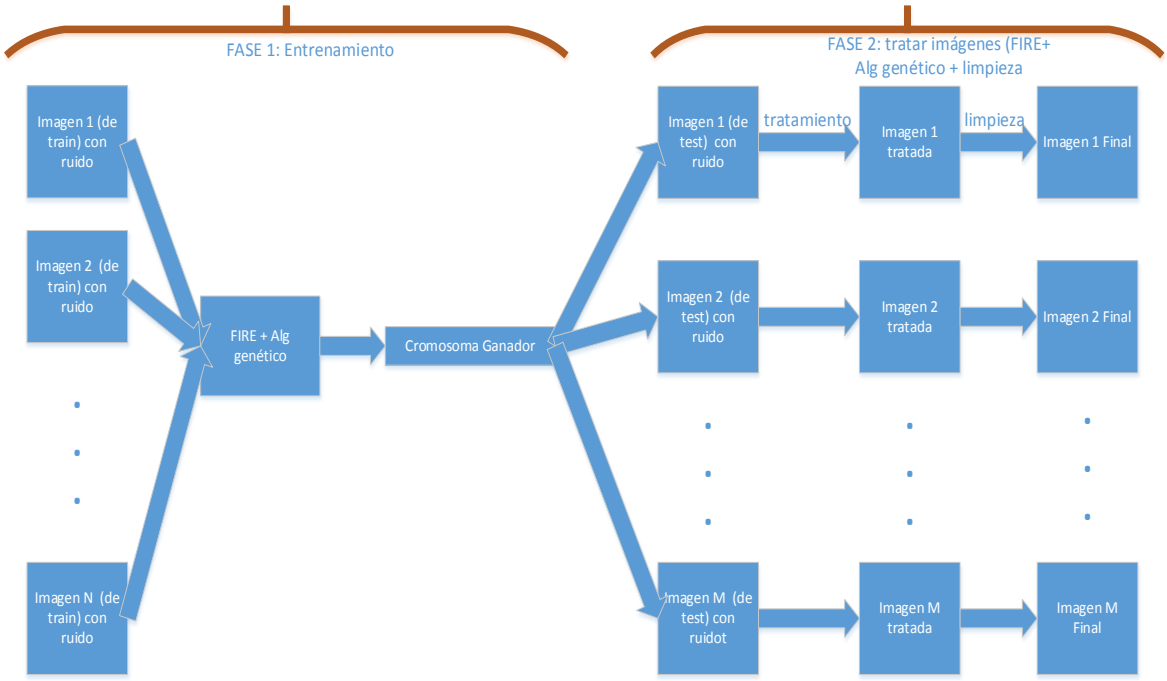
A la hora de la verdad no tiene sentido aplicar FIRE para aprender sobre una imagen con ruido teniendo la original sin ruido. Normalmente no se van a tener las imágenes con ruido y sin él. Con este sistema se va a lograr un cromosoma (con la mejor base de reglas) para un conjunto de imágenes de entrenamiento. Luego, esta base de reglas podrá ser aplicada para tratar cualquier imagen que reciba el sistema.

Con este último planteamiento se pretender ir un paso más allá y sacar más partido a las ventajas de un algoritmo genético. Para ello se ha modificado la estructura de todo el algoritmo para que en vez de limpiar una imagen con ruido, lo haga con varias. El objetivo de este nuevo algoritmo es que se pueda realizar una primera fase de aprendizaje con varias imágenes con ruido sal y pimienta para obtener un cromosoma (el mejor) sobre esas imágenes de entrenamiento. Toda la propuesta sigue el mismo procedimiento que el algoritmo original pero adaptado a más de una imagen.

Una vez tenemos el cromosoma ganador (el que mejor se ajusta a esas imágenes de entrenamiento) se aplica la segunda fase que se corresponde con la recuperación de la imagen y limpieza pero adaptada a cualquier imagen (las de entrenamiento y/o nuevas).

De esta forma, se ha construido un sistema capaz de poder entrenar con “X” imágenes de entrenamiento y en base al cromosoma obtenido, tratar cualquier otra imagen con ruido que se presente.

Para una mejor comprensión de este planteamiento, se va a proceder a realizar un esquema sobre su funcionamiento.



El propósito es conseguir sacar todo el potencial a los algoritmos genéticos y su posibilidad aprender sobre unos ejemplos. Con lo aprendido, actuar sobre cualquier caso que se nos presente creando un sistema general para tratar cualquier imagen en vez de quedarse con ajustar una imagen de manera individual.

Este algoritmo general se puede aplicar con cualquiera de las propuestas anteriores. De hecho este algoritmo general se emplea con cada una de las 4 versiones anteriormente para la siguiente sección de resultados.

4. RESULTADOS

En esta sección se van a exponer todas las pruebas realizadas, imágenes utilizadas, resultados obtenidos y sus posteriores análisis.

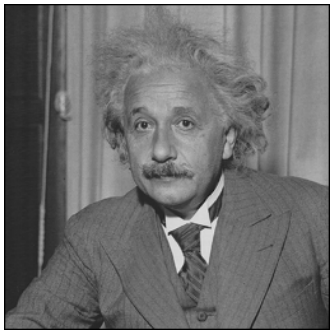
4.1 Marcos Experimental

En este apartado se van a detallar el conjunto de imágenes utilizadas y la configuración de los parámetros usados durante las pruebas.

A continuación se presentan las diferentes imágenes utilizadas para la realización de las pruebas. En total se han usado 27 imágenes de las cuales 7 han sido utilizadas para entrenamiento y el resto para test. Todas las imágenes que se han usado son de tamaño 256x256 píxeles.

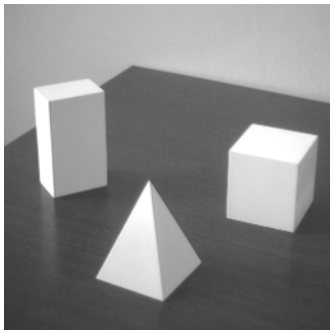
Imágenes de entrenamiento

5.2.10.png



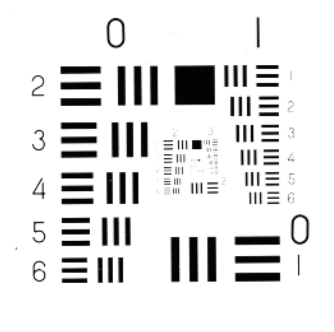
einstein.png

indor1.png



indor2.png

lena.png



test1.png

venice.png

Imágenes de test

4.1.png

4.1.06.png

4.2.05.png

4.2.06.png

aerial2.png

barche.png

car2.png

casa.png

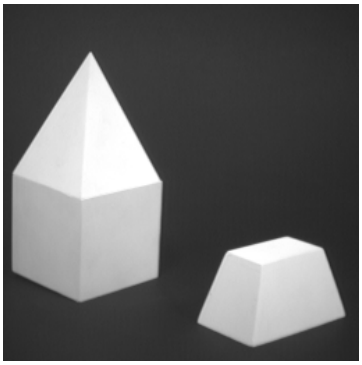
donna.png

elaine.512.png

goldhill.png

hedgebw.png

house.png



indor3.png

indor4.png

indor5.png

indor7.png

nat6.png

nat7.png

portofino.png

Se han realizado las pruebas con las mismas imágenes pero con diferentes densidades de ruido (0.1, 0.18 y 0.3). Para el desarrollo de las pruebas se han utilizado los algoritmos propuestos en el capítulo anterior:

1. Algoritmo general de limpieza + Implementación del algoritmo FIRE
2. Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP
3. Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP y LN
4. Algoritmo general de limpieza + Optimización de los parámetros del conjunto difuso LP para cada variable de entrada Δx_j

Para todas las propuestas la configuración de los parámetros es idéntica:

- Número de cadenas binarias de Substrings del cromosoma: M = 3

- Tamaño de ventana de vecindad: 3x3
- Fitness: $\frac{1}{MSE}$
- Parámetros de los conjuntos difusos: $a_0 = 128, a_1 = 120,\ a_2 = 40,\ b = 127\ y\ c = 0{,}5$. Este conjunto difuso siempre es incluido en todas las propuestas como cromosoma inicial.
- Cromosoma inicial: Cada propuesta tiene una representación diferente tal y como se ha explicado en el capítulo anterior. Son los siguientes:

1. Algoritmo general de limpieza + Implementación del algoritmo FIRE

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

2. Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5

3. Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP y LN

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5	127	120	40	127	0,5

4. Algoritmo general de limpieza + Optimización de los parámetros del conjunto difuso LP para cada variable de entrada Δx_j.

1	1	0	1	1	0	0	0	1	0	0	1	0	0	1	0	1	1	1	0	1	1	0	0

128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127

0,5	128	120	40	127	0,5	128	120	40	127	0,5	128	120	40	127	0,5

- Población inicial: 19 individuos aleatorios y el cromosoma inicial
- Selección de progenitores: método de la ruleta
- Cruce: Cruce por un punto para la parte binaria y PCBLX para la parte real
- Probabilidad de cruce: 0,9
- Mutación: sobre cada gen con una probabilidad de 0,05
- Tamaño de la siguiente generación: 20 individuos (los mejores)
- Condición de parada
    - Iteraciones máximas = 200
    - No se produzca mejora del Fitness del mejor cromosoma de 0,000001 durante 10 generaciones

Además de las propuestas, se han utilizado dos filtros para la realización de pruebas y comparación de resultados.

- Filtro de la Mediana: Tamaño de ventana de vecindad 3x3
- Filtro Median_base impulsive detector: Tamaño de ventana de vecindad 3x3 y umbral T=40

4.2 Resultados obtenidos

En este apartado vamos a ver los diferentes resultados para cada nivel de ruido. Por cada uno se mostrarán los resultados numéricos y visuales, así como un análisis de los mismos.

4.2.1 Resultados con densidad de ruido 0,1

A continuación se muestran los resultados visuales de algunas imágenes después de ser tratadas por los diferentes procedimientos para el nivel de ruido 0,1.

Debido a la gran cantidad de imágenes utilizadas en las pruebas, se muestran una selección de las imágenes de test por motivos de espacio.

Cada fila de la tabla 4-1 corresponde a los resultados de una imagen de test.

Las columnas de la tabla 4-1 son: la imagen original sin ruido, la imagen con nivel de ruido 0.1, las imágenes resultantes de cada uno de las propuestas (algoritmos 1-4) y las imágenes resultantes de los filtros de la Mediana y Median_base impulsive detector.

Esta estructura será igual para los otros dos niveles de ruidos expuestos en los apartados posteriores.

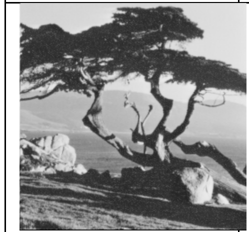
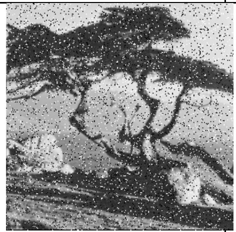
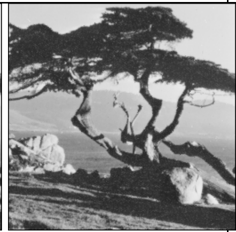
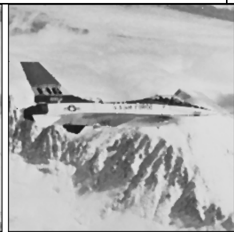
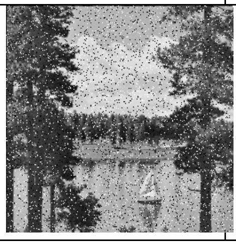
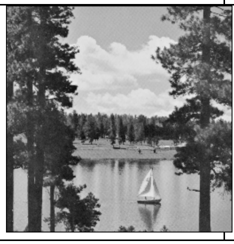
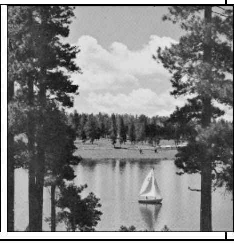
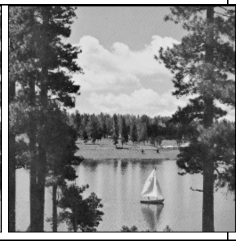
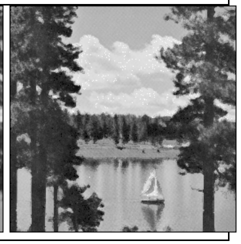
Imagen sin ruido	Imagen con ruido	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro Mediana	Filtro Median_base impulsive detector
							
							
							
							

Tabla 4-1: Resultados visuales para densidad de ruido 0,1

En la tabla 4-2 se muestran los resultados numéricos (MSE) para el nivel de ruido 0,1. Cada fila corresponde a los valores que surgen por cada una de las 27 imágenes mostradas en el marco experimental (tanto de entrenamiento como de test, 27 en total). Cada columna se corresponde con cada uno de los algoritmos propuestos y los dos filtros (Filtro de la Mediana y Filtro Median_base impulsive detector).

Estos algoritmos son:

- Algoritmo 1: Algoritmo general de limpieza + Implementación del algoritmo FIRE
- Algoritmo 2: Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP
- Algoritmo 3: Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP y LN
- Algoritmo 4: Algoritmo general de limpieza + Optimización de los parámetros del conjunto difuso LP para cada variable de entrada Δx_j

Cada celda se compone de dos elementos. El valor que aparece a la izquierda de cada celda, corresponde con el valor del Error Cuadrático Medio de cada algoritmo para cada imagen. Para cada fila, el elemento que mejor Error Cuadrático Medio tenga de todos (el más bajo) aparecerá en negrita. Además, el mejor de los cuatro algoritmos propuestos aparecerá subrayado.

En función de ese valor, en la parte derecha de cada celda, aparece un número entre paréntesis. Este valor sirve para realizar un ranking entre todos los valores de la fila. Estos valores van del 1 (el mejor) hasta número total de métodos, 6 (el peor de todos).

La última fila de la tabla contiene dos elementos. El valor superior realiza una media de estos rankings con el fin de proporcionar una idea de la calidad de cada uno de los métodos. El elemento inferior, corresponde con el número de veces que el método obtiene el mejor ranking.

	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro de la Mediana	Filtro Median_base impulsive detector
Lena	20,74 (3)	20,18 (2)	**19,56** (1)	22,01 (4)	71,37 (5)	185,09 (6)
5.2.10	66,01 (2)	68,79 (4)	**57,30** (1)	68,37 (3)	333,39 (5)	570,57 (6)
Einstein	**10,82** (1)	15,64 (4)	11,14 (2)	13,81 (3)	46,93 (5)	120,00 (6)
Indor1	**0,60** (1)	1,96 (4)	1,65 (3)	2,08 (5)	1,57 (2)	13,33 (6)
Indor2	**7,25** (1)	7,30 (2)	8,20 (3)	8,67 (4)	11,79 (5)	34,18 (6)
Test1	307,55 (4)	**171,37** (1)	271,88 (2)	305,47 (3)	492,11 (5)	521,95 (6)
Venice	88,82 (4)	**61,19** (1)	64,71 (2)	82,85 (3)	283,26 (5)	433,35 (6)
4.1.04	**10,25** (1)	16,18 (4)	10,87 (2)	13,62 (3)	21,55 (5)	79,84 (6)
4.1.06	**22,86** (1)	29,86 (4)	22,90 (2)	27,00 (3)	104,45 (5)	222,45 (6)
4.2.05	28,84 (2)	29,80 (3)	**24,51** (1)	29,84 (4)	122,57 (5)	220,96 (6)
4.2.06	37,50 (2)	38,51 (3)	**33,39** (1)	38,75 (4)	187,63 (5)	323,33 (6)
Aerial2	**23,20** (1)	29,24 (4)	23,49 (2)	28,67 (3)	104,45 (5)	288,90 (6)
Barche	**21,27** (1)	27,26 (4)	21,39 (2)	24,42 (3)	111,50 (5)	246,08 (6)
Car2	169,59 (3)	**106,79** (1)	151,06 (2)	174,69 (4)	312,78 (5)	378,95 (6)
Casa	**7,29** (1)	17,40 (4)	8,15 (2)	10,72 (3)	33,79 (5)	103,80 (6)
Donna	**5,49** (1)	8,99 (4)	5,98 (2)	7,43 (3)	25,54 (5)	108,47 (6)
Ealine.512	**14,96** (1)	17,41 (4)	15,40 (2)	17,29 (3)	57,79 (5)	158,58 (6)
Goldhill	**20,19** (1)	23,07 (3)	20,34 (2)	23,81 (4)	113,88 (5)	256,77 (6)
Hedgebw	28,30 (2)	31,54 (4)	**26,18** (1)	30,28 (3)	107,47 (5)	299,48 (6)
House	33,04 (2)	37,87 (4)	**31,72** (1)	36,57 (3)	182,88 (5)	338,29 (6)
Indor3	8,53(3)	**6,98** (1)	8,47 (2)	9,12 (4)	10,51 (5)	25,43 (6)
Indor4	**5,56** (1)	8,80 (4)	8,28 (3)	7,68 (2)	14,85 (5)	49,95 (6)
Indor5	**7,35** (1)	13,37 (4)	8,68 (2)	10,95 (3)	27,20 (5)	78,37 (6)
Indor7	**4,40** (1)	8,96 (4)	5,30 (2)	7,52 (3)	17,10 (5)	78,69 (6)
Nat6	**1,24** (1)	3,74 (4)	1,90 (2)	2,70 (3)	5,92 (5)	40,27 (6)
Nat7	**27,21** (1)	35,03 (4)	27,42 (2)	33,18 (3)	107,22 (5)	215,66 (6)
Portofino	23,52 (2)	28,07 (4)	**22,02** (1)	26,68 (3)	144,97 (5)	276,71 (6)
MEDIA (Ranking)	1,67 (16)	3,30 (4)	1,85 (7)	3,30 (0)	4,69 (0)	6,00 (0)

Tabla 4-2: Resultados para densidad de ruido 0,1

A la vista de los resultados numéricos y visuales hay varios aspectos que merecen ser comentados.

Para el nivel de ruido 0,1 se observa como el algoritmo 1 y 3 son claramente mejor al resto con una media de ranking de 1,67 y 1,85 respectivamente. A pesar de tener un ranking parecido el algoritmo 1 es mejor en 16 de las 27 imágenes utilizadas mientras que el algoritmo 3 es mejor en 7. También es destacable que la diferencia entre ambos algoritmos (error cuadrático medio) es muy pequeña para cada imagen.

Es curioso como el algoritmo 2 y 4 tienen exactamente la misma media de ranking. A pesar de ello, el algoritmo 2 es el mejor en 4 de las imágenes. Ambos no consiguen ser nunca el mejor método.

También se puede ver como entre los dos filtros el de la mediana actúa mucho mejor que el Median_base impulsive detector aunque ambos obtienen mucho peores resultados que cualquiera de las propuestas y no consiguen ser el mejor en ninguna ocasión.

Por último cabe comentar la curiosidad de que el algoritmo que mejor funcione sea el 1, a pesar de ser el que menos parámetros de aprendizaje tiene por lo que pasa este nivel de ruido se puede concluir que el hecho de añadir más parámetros al cromosoma para que se desarrolle con más libertad y se obtengan mejores resultados no se cumple.

4.2.2 Resultados con densidad de ruido 0,18

En la tabla 4-3 se muestran los resultados visuales de algunas imágenes con densidad de ruido 0,18 después de ser tratadas por los diferentes procedimientos.

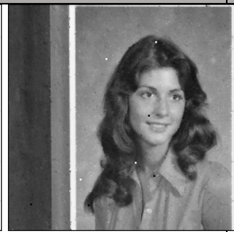
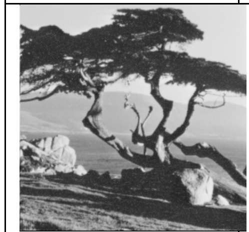
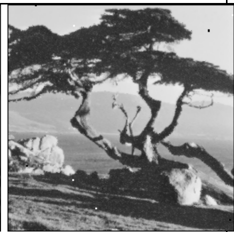
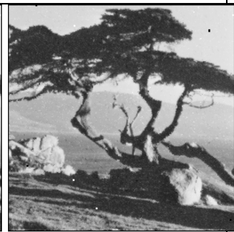
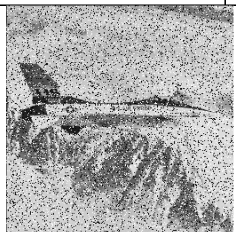
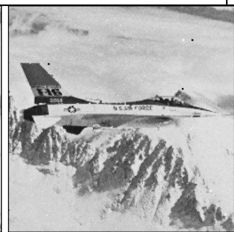
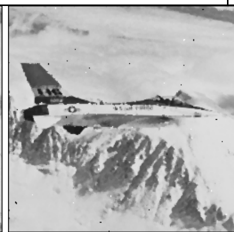
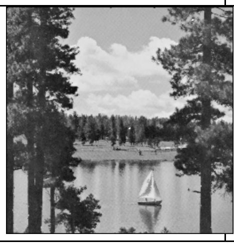
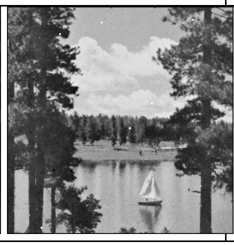
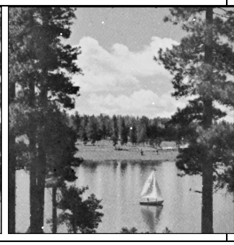
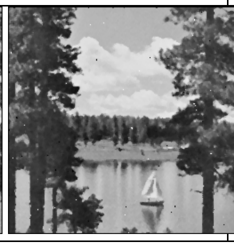
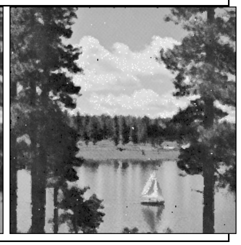
Imagen sin ruido	Imagen con ruido	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro Mediana	Filtro Median_base impulsive detector
							
							
							
							

Tabla 4-3: Resultados visuales para densidad de ruido 0,18

En la tabla 4-4 se muestran los resultados para el nivel de ruido 0,18.

	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro de la Mediana	Filtro Median_base impulsive detector
Lena	**37,57** (1)	69,94 (4)	50,02 (2)	63,42 (3)	108,34 (5)	215,01 (6)
5.2.10	**132,97** (1)	159,69 (4)	134,46 (2)	154,66 (3)	402,32 (5)	615,29 (6)
Einstein	**26,83** (1)	48,80 (3)	36,14 (2)	50,63 (4)	83,79 (5)	148,14 (6)
Indor1	**3,58** (1)	23,85 (4)	10,91 (2)	24,75 (5)	20,71 (3)	30,82 (6)
Indor2	**7,73** (1)	26,87 (4)	13,56 (2)	26,41 (3)	28,56 (5)	50,76 (6)
Test1	467,47 (4)	**395,31** (1)	455,64 (3)	430,26 (2)	682,72 (5)	719,09 (6)
Venice	148,18 (3)	140,50 (2)	**140,38** (1)	155,09 (4)	351,04 (5)	502,48 (6)
4.1.04	**16,71** (1)	42,65 (3)	25,25 (2)	42,65 (3)	52,40 (5)	105,66 (6)
4.1.06	**63,89** (1)	88,15 (4)	65,95 (2)	83,83 (3)	165,97 (5)	275,75 (6)
4.2.05	**54,90** (1)	71,58 (3)	59,80 (2)	73,80 (4)	167,55 (5)	264,25 (6)
4.2.06	**78,33** (1)	103,84 (4)	85,33 (2)	102,67 (3)	241,86 (5)	367,47 (6)
Aerial2	**50,53** (1)	88,01 (3)	66,81 (2)	89,16 (4)	146,41 (5)	310,61 (6)
Barche	**55,05** (1)	77,78 (3)	60,50 (2)	82,99 (4)	164,81 (5)	287,76 (6)
Car2	200,51 (2)	**196,23** (1)	202,43 (3)	225,08 (4)	355,05 (5)	419,22 (6)
Casa	**24,29** (1)	48,50 (4)	34,43 (2)	47,26 (3)	72,35 (5)	135,52 (6)
Donna	**18,43** (1)	39,89 (4)	26,50 (2)	39,34 (3)	62,31 (5)	146,38 (6)
Ealine.512	**31,45** (1)	54,06 (3)	41,08 (2)	56,82 (4)	92,46 (5)	182,62 (6)
Goldhill	**44,66** (1)	66,02 (4)	47,41 (2)	65,14 (3)	150,64 (5)	277,70 (6)
Hedgebw	**53,93** (1)	91,28 (4)	68,40 (2)	77,71 (3)	152,83 (5)	332,88 (6)
House	**78,28** (1)	100,76 (3)	81,97 (2)	101,89 (4)	244,74 (5)	384,26 (6)
Indor3	**9,03** (1)	32,31 (4)	18,49 (2)	28,02 (3)	41,44 (5)	58,64 (6)
Indor4	**11,81** (1)	33,37 (3)	24,99 (2)	36,57 (4)	44,31 (5)	77,37 (6)
Indor5	**28,99** (1)	51,32 (3)	41,83 (2)	58,08 (4)	72,10 (5)	118,56 (6)
Indor7	**18,00** (1)	36,46 (3)	22,08 (2)	39,84 (4)	40,01 (5)	93,69 (6)
Nat6	**11,74** (1)	37,96 (4)	19,92 (2)	31,06 (3)	38,86 (5)	72,98 (6)
Nat7	**55,77** (1)	85,31 (3)	67,87 (2)	87,55 (4)	161,18 (5)	257,47 (6)
Portofino	**49,17** (1)	74,85 (4)	56,23 (2)	72,87 (3)	188,95 (5)	307,71 (6)
MEDIA (Ranking)	1,22 (24)	3,30 (2)	2,04 (1)	3,48 (0)	4,85 (0)	6,00 (0)

Tabla 4-4: Resultados para densidad de ruido 0,18

En esta ocasión se pueden comentar varios detalles que merecen ser citados a la vista de los resultados numéricos y visuales.

Se observa como el algoritmo 1 es sin ninguna duda el mejor. Con una media de 1,22 y siendo el mejor en 24 de las 26 imágenes, es decir, en todas menos dos.

El algoritmo 3 queda asentado como segundo método. A pesar que solo consigue ser el mejor en una imagen, sus diferencias de error cuadrático medio con el algoritmo 3 para cada imagen son relativamente pequeñas.

Los algoritmos 2 y 4 tienen una media muy pareja de 3,30 y 3,48 respectivamente. Aunque no tienen una media muy alta, el algoritmo 2 consigue ser el mejor método en 2 imágenes.

También se puede ver como entre los dos filtros, el de la mediana actúa mejor que el Median_base impulsive detector, aunque ambos lo hacen mucho peor que cualquiera de las propuestas, como se puede observar en el ranking.

Para el nivel de ruido 0,18 y a vista de los resultados se puede concluir que el incluir más parámetros en el cromosoma con el fin de dejarle más autonomía en la búsqueda de una mejor solución, no garantiza que acabe siendo así ya que el algoritmo 1 es claramente el mejor.

4.2.3 Resultados con densidad de ruido 0,3

A continuación, en la tabla 4-5, se muestran los resultados visuales de algunas imágenes con densidad de ruido 0,3 después de ser tratadas por los diferentes procedimientos.

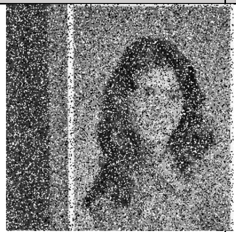
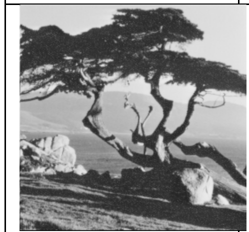
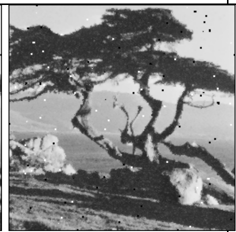
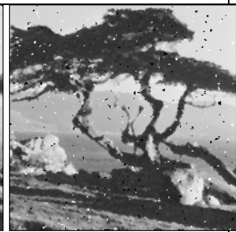
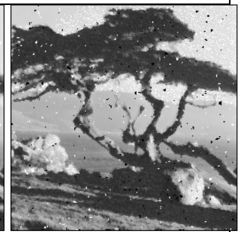
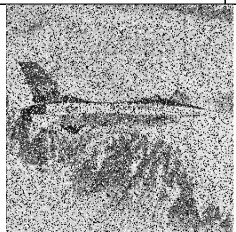
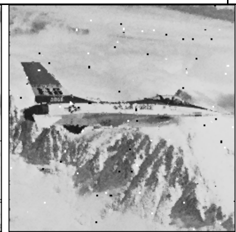
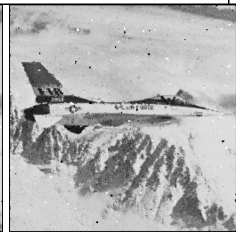
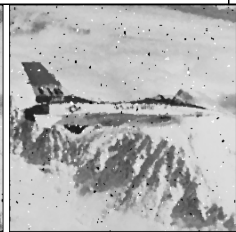
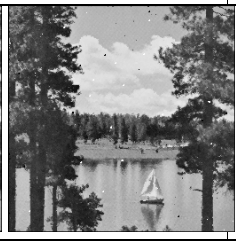
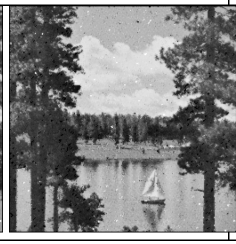
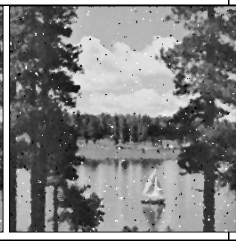
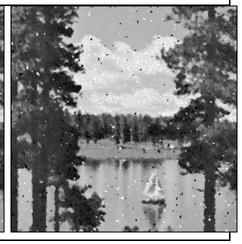
Imagen sin ruido	Imagen con ruido	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro Mediana	Filtro Median_base impulsive detector
							
							
							
							

Tabla 4-5: Resultados visuales para densidad de ruido 0,3

En la siguiente tabla 4-6 se muestran los resultados para el nivel de ruido 0,3.

	Algoritmo 1	Algoritmo 2	Algoritmo 3	Algoritmo 4	Filtro de la Mediana	Filtro Median_base impulsive detector
Lena	**101,25** (1)	115,38 (2)	184,74 (3)	201,76 (4)	364,05 (5)	460,12 (6)
5.2.10	**271,21** (1)	282,20 (2)	375,00 (4)	373,56 (3)	696,59 (5)	874,94 (6)
Einstein	86,69 (2)	**84,37** (1)	175,14 (3)	194,05 (4)	324,19 (5)	374,10 (6)
Indor1	**20,65** (1)	34,09 (2)	92,98 (3)	138,53 (4)	183,83 (5)	191,09 (6)
Indor2	**49,63** (1)	50,68 (2)	112,98 (3)	134,05 (4)	243,31 (5)	266,84 (6)
Test1	1029,66 (4)	**860,33** (1)	903,23 (2)	978,29 (3)	1375,72 (5)	1409,72 (6)
Venice	282,89 (2)	**259,03** (1)	341,45 (3)	358,47 (4)	634,54 (5)	786,13 (6)
4.1.04	**52,31** (1)	79,93 (2)	149,04 (3)	163,69 (4)	261,86 (5)	308,29 (6)
4.1.06	**143,47** (1)	174,55 (2)	236,93 (4)	233,52 (4)	459,10 (5)	551,94 (6)
4.2.05	161,31 (2)	**154,88** (1)	228,66 (4)	213,16 (3)	427,26 (5)	520,28 (6)
4.2.06	176,98 (2)	**167,11** (1)	245,26 (3)	253,87 (4)	491,70 (5)	603,55 (6)
Aerial2	**104,32** (1)	114,33 (2)	196,31 (3)	217,72 (4)	373,90 (5)	505,24 (6)
Barche	111,05 (2)	**110,03** (1)	182,42 (3)	218,77 (4)	391,43 (5)	492,51 (6)
Car2	293,39 (2)	**268,61** (1)	339,29 (3)	398,89 (4)	567,20 (5)	623,85 (6)
Casa	80,04 (2)	**77,34** (1)	163,69 (4)	161,92 (3)	304,10 (5)	357,57 (6)
Donna	**45,95** (1)	63,77 (2)	112,71 (3)	130,62 (4)	267,12 (5)	354,33 (6)
Ealine.512	**69,55** (1)	89,88 (2)	172,34 (3)	176,35 (4)	296,36 (5)	373,37 (6)
Goldhill	**117,23** (1)	123,31 (2)	176,14 (3)	222,49 (4)	379,31 (5)	480,17 (6)
Hedgebw	**131,63** (1)	162,65 (2)	239,05 (3)	227,66 (4)	432,70 (5)	593,21 (6)
House	202,58 (2)	**196,98** (1)	275,79 (3)	275,96 (4)	501,87 (5)	616,66 (6)
Indor3	**38,78** (1)	84,89 (2)	127,01 (3)	143,05 (4)	259,69 (5)	280,55 (6)
Indor4	**42,89** (1)	60,28 (2)	131,34 (3)	145,50 (4)	240,95 (5)	271,07 (6)
Indor5	**70,41** (1)	75,05 (2)	154,84 (4)	152,70 (3)	270,66 (5)	310,09 (6)
Indor7	**49,24** (1)	52,01 (2)	133,19 (3)	137,26 (4)	229,11 (5)	271,90 (6)
Nat6	**40,56** (1)	63,82 (2)	130,81 (3)	150,70 (4)	223,07 (5)	257,76 (6)
Nat7	144,56 (2)	**125,71** (1)	217,20 (4)	210,86 (3)	401,51 (5)	477,00 (6)
Portofino	**125,01** (1)	141,10 (2)	206,81 (3)	229,27 (4)	427,03 (5)	522,48 (6)
MEDIA (Ranking)	1,41 (17)	1,63 (10)	3,19 (0)	3,78 (0)	5,00 (0)	6,00 (0)

Tabla 4-6: Resultados para densidad de ruido 0,3

Para la densidad de ruido 0,3 tenemos varios detalles a resaltar.

Se observan unos resultados bastante parejos entre el algoritmo 1 y 2. Tienen una media de ranking de 1,41 y 1,63 respectivamente. Entre los dos algoritmos ocupan la primera posición del ranking para todas las imágenes, siendo el algoritmo 1 el mejor en 17 imágenes y el algoritmo 2 en 10.

Se puede apreciar como el algoritmo 3 obtiene mejores resultados que el algoritmo 4. Ambos algoritmos ocupan el tercer y cuarto lugar en el ranking como se puede apreciar en la tabla 4-6.

Los filtros de la mediana y median-base impulsive detector funcionan mucho peor que las propuestas. Entre ellos, el de la mediana obtiene mucho mejores resultados que el median-base impulsive detector.

5. CONCLUSIONES

Para terminar se puede afirmar que se ha logrado cumplir los objetivos de este trabajo con éxito. Se ha entendido y conseguido replicar la propuesta original FIRE. Además se han ideado nuevas propuestas de mejora y se ha creado un sistema general de entrenamiento-test que sirve para tratar cualquier imagen a posteriori. Los resultados para los algoritmos 2, 3 y 4 (las nuevas propuestas) no han sido los esperados porque se supone que deberían mejorar al algoritmo 1 (ya que el algoritmo 1 debería ser un caso partícular de las propuestas). Esto es debido a que la base de reglas que representa el cromosoma ganador del algoritmo 1 es de mejor calidad que los obtenidos con las propuestas. La explicación a todo esto es que al añadir más parámetros a nuestro cromosoma, su espacio de búsqueda es mucho mayor y por consiguiente, hace que se pierda en él. Para todas estas propuestas se ha realizado un estudio en el que se muestra la calidad de las propuestas que se han elaborado y que demuestran el éxito de las mismas.

Tras la realización del estudio experimental y en base a los resultados obtenidos en las pruebas para cada uno de los niveles de ruido, se pueden resaltar las siguientes lecciones aprendidas:

Para un nivel de ruido bajo (0.1), la propuesta que mejor funciona es el algoritmo 1 (Algoritmo general de limpieza + algoritmo FIRE original). Aunque es el mejor método en 16 de las 27 imágenes, el algoritmo 3 (Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP y LN) posee una media de ranking bastante similar. Si el algoritmo 1 tiene un 1,67 de media, el algoritmo 3 posee 1,85. A pesar de ello, la realidad es que es algoritmo 1 es el mejor método en líneas generales. Los algoritmos 2 y 4 obtienen unos resultados bastante similares entre ellos pero muy discretos en comparación a los algoritmos 1 y 3. Aunque a priori se pudiera pensar que añadiendo más parámetros al cromosoma con el fin de otorgar más libertad de aprendizaje al algoritmo genético, no se obtiene un mejor patrón de eliminación del ruido.

Para un nivel de ruido intermedio de densidad 0.18, el algoritmo 1 sigue siendo el mejor método. Si para el nivel anterior de ruido era el mejor en 16 imágenes, en este, es mejor en 24 de las 26 imágenes. A pesar de ser el mejor con tanta contundencia al ver los datos, se puede destacar que las diferencias de error con el segundo mejor método (el algoritmo 3) para cada imagen son mínimas. Aunque el algoritmo 3 vuelve a ser el segundo mejor método, su media de ranking es un poco peor que para el nivel de ruido 0,1 y pasa de ser el mejor en 7 imágenes a 1. Si el algoritmo 3 funcionaba casi tan bien como el algoritmo 1 para un nivel de ruido 0.1, se puede apreciar como para un nivel de ruido superior no lo hace tan bien ya que pasa a ser el segundo mejor método al tercero. Por lo tanto, para un nivel de ruido intermedio tampoco se cumple que incluir más

parámetros en el cromosoma nos asegure mejores resultados ya que el algoritmo 1 vuelve a ser el mejor.

Para un nivel de ruido alto (0,3), el algoritmo 1 vuelve a tener los mejores resultados. Hay que decir que a pesar de ser el mejor, el algoritmo 2 (Algoritmo general de limpieza + Ajuste de los parámetros del conjunto difuso LP) obtiene unos resultados muy similares. La media de ranking del algoritmo 1 es de 1,41 por 1,63 del algoritmo 2. En este caso el algoritmo 1 es el mejor en 17 imágenes por 10 del algoritmo 2. Se puede destacar que el hasta ahora siempre segundo mejor método (el algoritmo 3) pasa a la tercera posición y no consigue ser el mejor en ninguna imagen. Con esto se puede afirmar que a medida que aumenta la densidad de ruido, el algoritmo 3 obtiene peores resultados mientras que con el algoritmo 2 ocurre justo lo contrario.

Un aspecto muy destacable es que el algoritmo 4 (Algoritmo general de limpieza + Optimización de los parámetros del conjunto difuso LP para cada variable de entrada) obtiene en general los peores resultados entre las propuestas. Se desarrolló con idea de obtener los mejores resultados ya que se le incluían 40 nuevos parámetros al cromosoma con el fin de darle una flexibilidad máxima al algoritmo genético y obtener mejores soluciones. Ha quedado demostrado en vista de las pruebas que el dotar de mayor libertad y añadir parámetros al cromosoma no me garantiza obtener unos resultados mejores.

El filtro de la mediana ha sido siempre un método muy utilizado ante un problema de ruido sal y pimienta en imágenes por sus buenos resultados. En los distintos niveles de ruido probados, las propuestas elaboradas en este proyecto siempre trabajan mejor que los filtros usados para la comparación.

6. LÍNEAS FUTURAS

Vistos los resultados obtenidos, se han considerado una serie de posibles vías de estudio que pueden ser interesantes de desarrollar.

En vista de que el algoritmo 1 (FIRE original) consigue obtener un cromosoma ganador de mejor calidad al generado por las propuestas, existen dos alternativas:

1. Introducir el cromosoma ganador del algoritmo 1 como cromosoma inicial y ejecutar después cualquiera de nuestras propuestas. Con esto conseguimos que al menos nuestra peor solución sea la obtenida con el algoritmo 1. Este planteamiento tiene una crítica porque obliga a ejecutar el algoritmo 1 y además la propuesta elegida, por lo que el tiempo de ejecución aumenta considerablemente.
2. Idear una nueva estructura de nuestros algoritmos de tal forma que primero se realice el algoritmo 1 (cromosoma en binario) y una segunda etapa compuesta por un algoritmo genético cuyo objetivo sea aprender los mejores parámetros de los conjuntos difusos utilizando en esta fase la base de reglas aprendida en la primera etapa.

Durante el proyecto se han trabajado con imágenes de 256x256 píxeles tanto en train y test. Una posible prueba es ver cómo se comporta el sistema si cambiamos los tamaños de las imágenes. Es decir, entrenar con un determinado tamaño y luego utilizar imágenes de test con un tamaño mayor o menor. Esto puede ser muy interesante, ya que si se consiguen buenos resultados al eliminar el ruido, se pueden reducir mucho los tiempos de entrenamiento y/o de test.

Además, otro posible estudio pudiera ser el ver cómo se comporta el algoritmo FIRE ante imágenes con otros tipos de ruido distintos al ruido impulsivo Sal y Pimienta.

Por otra parte, el algoritmo FIRE utiliza siempre ventanas de vecindad 3x3. Una posible modificación puede ser probar a aumentar este tamaño de vecindad a (5x5, 7x7, etc) y comprobar si su funcionamiento es mejor. Habría que modificar también el número de entradas en función de ese tamaño de vecindad.

7. BIBLIOGRAFÍA

1. FIRE operators for image processing
   Fabrizio Russo. Fuzzy sets and System 103 (1999) 265-275

2. Introducción al procesamiento de imagen. Visión Artificial
   Daniel Paternain, Edurne Barrenechea. Universidad Pública de Navarra

3. Filtrado y realzado de imagen. Visión Artificial
   Daniel Paternain, Edurne Barrenechea. Universidad Pública de Navarra

4. Comprensión y reducción de imágenes. Visión Artificial
   Daniel Paternain, Edurne Barrenechea. Universidad Pública de Navarra

5. Basic Concepts. Ingeniería del Conocimiento
   Humberto Bustince. Universidad Pública de Navarra

6. Aspectos avanzados de la Inteligencia Artificial. Extensiones de los conjuntos difusos en la representación del conocimiento. Ingeniería del Conocimiento
   Humberto Bustince. Universidad Pública de Navarra

7. Introducción a los algoritmos genéticos. Computación
   Fco. Javier Fernandez. Universidad Pública de Navarra

8. Algoritmos genéticos con codificación binaria. Computación
   Fco. Javier Fernandez. Universidad Pública de Navarra

9. Algoritmos genéticos con codificación general. Computación
   Fco. Javier Fernandez. Universidad Pública de Navarra

10. Filtrado de imágenes
    http://dmi.uib.es/~ygonzalez/VI/Material_del_Curso/Teoria/Tema5_Filtrado.pdf

11. Procesamiento digital de imágenes
    http://verona.fi-p.unam.mx/boris/teachingnotes/Introduccion.pdf

12. Filtros. Conceptos generales
    http://alojamientos.us.es/gtocoma/pid/tema3-1.pdf

13. Teoría de conjuntos difusos y lógica difusa
http://www.lcc.uma.es/~eva/aic/apuntes/fuzzy.pdf

14. Conceptos fundamentales de la lógica difusa
http://www.tesisenred.net/bitstream/handle/10803/6887/04Rpp04de11.pdf?sequence=4

15. Introducción a la lógica difusa
http://es.slideshare.net/mentelibre/logica-difusa-introduccion

16. Algoritmos genéticos
http://www.sc.ehu.es/ccwbayes/docencia/mmcc/docs/temageneticos.pdf