# Algoritmos

A continuación, se explica el desarrollo de los algoritmos utilizados.

## PSO

Texto

Descripción generada automáticamente

Ilustración 15 Comienzo PSO

En la ilustración 15 se muestran los imports de este algoritmo y dos variables globales que definen el comportamiento de la velocidad máxima y mínima para cada individuo del algoritmo en este caso 4 y -4.

Interfaz de usuario gráfica, Texto, Aplicación

Descripción generada automáticamente

Ilustración 16 Constructor PSO

En la ilustración 16 comienza la clase que define el PSO y su constructor, sus parámetros son los siguientes:

* n: número de individuos.
* function: función que se aplica en el algoritmo.
* lb: límite inferior del espacio de búsqueda.
* ub: límite superior del espacio de búsqueda.
* dimension: dimensión del espacio de solución (r).
* iteration: número de iteraciones.
* numeroColores: número de colores de la nueva imagen.
* pintor: booleano que se usa para saber si pintamos imagen al final o no (Es el argumento pintaImagen del programa).
* w: parámetro inercia.
* c1: parámetro cognitivo.
* c2: paramero social.
* imagen: ruta de la imagen a procesar por el algoritmo.

Texto

Descripción generada automáticamente

Ilustración 17 Inicialización PSO

En la anterior ilustración se ve como se inicializa cada uno de los parámetros.

Gracias al constructor de intelligence se inicializan posiciones, mejores posiciones y mejor fitness, todo a 0 en un principio (Ilustración 18)

También se generan números aleatorios gracias a la librería numpy de manera uniforme y aleatoria, esto da las primeras posiciones de cada individuo.

Se inicializa el vector de velocidad a 0 y las mejores posiciones de los individuos que son las primeras que obtuvimos antes.

Se calcula el fitness actual de todos los individuos y se inicializa el mejor fitness con esos valores.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamentePor último, se inicializa la mejor posición global llamando a nuestra función objetivo (MSE, MAE, SSIM o MSSIM) para cada una de las posiciones de cada individuo, el resultado de esta función es el fitness de cada individuo por lo que con argmin() obtenemos el menor resultado, es decir, la mejor posición de todos los individuos.

Ilustración 18 Constructor de Intelligence

Texto

Descripción generada automáticamente

Ilustración 19 Comienzo bucle PSO

En la ilustración 19 comienza el bucle del PSO, es decir, las instrucciones que va a realizar el algoritmo en cada una de las iteraciones.

Comienza por calcular r1 y r2 llamando a numpy para obtener una matriz de números aleatorios entre 0 y 1 (np.random.rand()) de tamaño (n, numeroColores, dimension), donde para cada individuo n se generan numeroColores valores aleatorios en un espacio de dimension dimensiones, es decir, si tenemos 10 individuos y una paleta de colores de 64 el resultado sería una matriz de tamaño (10, 64, 3) y sus valores serian números aleatorios entre 0 y 1.

En la línea 62 se calcula la velocidad siguiendo la fórmula de la velocidad del PSO definida en la página 19, después se ajusta esta velocidad con clip() a los valores permitidos (entre -4 y 4).

Se ajusta la posición de las partículas sumando la velocidad, se acota esta posición a los límites del espacio permitidos.

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

Ilustración 21 Actualización mejor solución particular PSO

En la ilustración 21 se muestra la actualización de los valores del fitness actual, ejecutando la función objetivo para todos los individuos.

Después se muestra un bucle que recorre todos los individuos que tenga el algoritmo en el cual se comprueba si el fitness actual es mejor que los mejores valores de fitness guardados hasta el momento, si esto se cumple se actualiza la mejor posición del individuo y el mejor fitness del individuo.

Ilustración 22 Actualización mejor solución global PSO

Texto

Descripción generada automáticamentePor último, según como se muestra en la ilustración 22 se actualiza la mejor solución global cogiendo el índice del individuo cuyo fitness sea menor gracias a la librería Numphy y su función argmin()

Captura de pantalla de un celular

Descripción generada automáticamenteSe actualiza el mejor fitness hallado y se imprime.

Ilustración 23 Fin PSO

En esta ilustración se muestra el final del PSO en el que se guarda la mejor solución global obtenida por el algoritmo, y se genera una imagen cuantizada con esa mejor solución (mejor paleta de colores) para luego pintarla. Esta última generación de cuantizada no hace falta si no se pinta.

## Luciérnagas

De igual manera que en el anterior algoritmo se comienza con el constructor.



Ilustración 24 Constructor Luciérnagas

En esta ilustración se muestra el constructor cuyos parámetros son:

* n: número de partículas
* function: función a optimizar
* lb: límite inferior del espacio (0 para imágenes)
* ub: límite superior del espacio (255 para imágenes)
* dimension: dimensiones del espacio
* iteration: número de iteraciones
* numeroColores: número de colores de la nueva imagen
* pintor: booleano que se usa para saber si pintamos imagen al final o no.
* csi: atracción mutua
* psi: Coeficiente de absorción de la luz del medio
* alpha0: valor inicial del parámetro aleatorio alpha
* alpha1: valor final del parámetro aleatorio alpha
* norm0: primer parámetro para una distribución normal (Gaussiana)
* norm1: segundo parámetro para una distribución normal (Gaussiana)
* imagen: ruta de la imagen a procesar por el algoritmo

Captura de pantalla con la imagen de una pantalla

Descripción generada automáticamente con confianza media

Ilustración 25 Inicio luciérnagas

Se continúa inicializando intelligence, la población de individuos y las mejores posiciones de cada individuo de la misma manera que en el anterior algoritmo.

Por último, se calculan todos los fitness, fitnessP representa los mejores valores de fitness (mejores posiciones) que se han encontrado y fitnessA el fitness actual, como se esta inicializando no importa que no sea del todo correcto ya que en un principio tanto la mejor posición como el mejor fitness coincide con la primera posición y el primer fitness hallado.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

Ilustración 26 Comienzo del bucle luciérnagas

En esta última ilustración se comienza el bucle del algoritmo, y se calcula alpha que será necesario para la función de mover individuos.

Alpha = valor inicial del parámetro alpha + (valor final del parámetro alpha) \* exponcial(-numero iteracion).

Texto

Descripción generada automáticamenteEn esta ilustración se muestran dos bucles anidados que recorren cada individuo y por cada individuo se recorre de nuevo cada individuo en el algoritmo por lo que se compara cada individuo con todos y cada uno de los individuos. Se hace para que si el fitness actual de un individuo i es mayor el que el de un individuo j se mueve este individuo i hacia el individuo j ya que el individuo j es más atractivo, si no cumple esa condición se mueve aleatoriamente por el espacio.

Ilustración 27 Movimientos de luciernagas

Estas operaciones se realizan para mover cada luciérnaga a las que son más brillantes que ellas para realizar una explotación global.

Texto

Descripción generada automáticamenteEn esta ilustración se ve como se acota la posición al rango permitido y se transforma a lista (de igual manera que en el algoritmo anterior).

Ilustración 28 Cálculos Luciérnagas

Con estos pasos concluye el movimiento de luciérnagas por lo que se puede empezar a calcular el fitness actual (fitnessA) de manera igual que en el PSO, para cada individuo se llama a la función objetivo para obtener el fitness.

Ilustración 29 Actualización posición y fitness luciérnagas

Texto

Descripción generada automáticamente

En la ilustración 29 se muestra cómo se actualiza tanto la mejor solución particular de cada individuo y su mejor fitness.

Texto

Descripción generada automáticamenteTambién se actualiza la mejor solución global y el mejor fitness, por último, se imprime este valor.

Ilustración 30 Actualización mejor solución global luciérnagas

Captura de pantalla de computadora

Descripción generada automáticamente

En esta última ilustración se muestra cómo se mueven las luciérnagas siguiendo las fórmulas que están en los comentarios.

Ilustración 31 Función mover luciérnaga

Para el cálculo de la distancia (r) se utiliza la biblioteca numpy, en este caso se calcula la distancia euclidiana que es la distancia recta entre dos puntos en el espacio.

Esta función devuelve las nuevas posiciones de cada individuo.

## Lobos

Ilustración 32 Inicialización Lobos

Captura de pantalla de computadora

Descripción generada automáticamente

Como en el resto de los algoritmos se empieza con el constructor del algoritmo e inicializando la población.

En este algoritmo los parámetros son:

* n: numero de individuos
* function: función del algoritmo
* lb: limite inferior del espacio de búsqueda
* ub: limite superior del espacio de búsqueda
* dimension: dimensión del espacio
* iteration: número de iteraciones

Después de inicializar intelligence y la población, se calculan los valores de fitness actual de cada individuo y se buscan los mejores lobos (alfa beta y delta ABD), alpha representa el individuo con mejor posición por lo cual se asigna Gbest (mejor posición global) con esta posición.

Texto

Descripción generada automáticamente

Ilustración 33 Get mejores lobos

Esta función retorna los tres mejores individuos de todos los individuos del algoritmo y el fitness del mejor individuo. Para ello primero inicializa result como una lista vacía, después se calcula el fitness de cada individuo y se guarda en una lista junto con su índice. Después se ordena esta lista en orden ascendente.

Texto

Descripción generada automáticamentePor último, se recorren los tres mejores individuos para añadirlos a result, se selecciona de la lista de individuos el individuo cuyo índice coincide con el índice almacenado en la lista fitness y se añade a la lista result el fitness del mejor individuo para recuperarlo facilmente.

Ilustración 34 Inicio bucle Lobos

En esta ilustración se muestra el inicio del bucle del algoritmo y el cálculo del parámetro a siguiendo esta ecuación:

Texto

Descripción generada automáticamente

Ilustración 35 Calculo de aleatorios, A y C

En esta ilustración se muestra el cálculo de números aleatorios para hallar A y C de cada individuo siguiendo las ecuaciones 1 y 2 de la página 31.

Texto

Descripción generada automáticamente

Ilustración 36 Cálculo de D

Se calculan las D de cada individuo alfa beta y delta siguiendo las ecuaciones de la página 33.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

Ilustración 37 Cálculo de X para cada lobo

Texto

Descripción generada automáticamenteEn esta ilustración se muestra el cálculo de X para cada individuo siguiendo las ecuaciones de la página 33.

Ilustración 38 Final bucle lobos

Se ajustan las posiciones de los individuos siguiendo la ecuación de la página 33.

Se acotan las nuevas posiciones de cada individuo a los límites permitidos.

Llamando de nuevo a la función getABD() se obtienen los tres mejores individuos y el fitness del alfa.

Después se comprueba si el fitness actual del mejor individuo es mejor que el mejor fitness almacenado, en caso verdadero se actualiza la mejor posición global con la posición del individuo alfa y se actualiza el mejor fitness.

Por último, se guarda el mejor fitness y se imprime.

## Abejas

Ilustración 39 Constructor abejas

Aquí se muestra el constructor de abejas, estos son sus parámetros:

* n: número de individuos
* function: función
* lb: límite inferior del espacio
* ub: límite superior del espacio
* dimension: dimensión del espacio
* Pantalla de computadora con letras

  Descripción generada automáticamente con confianza mediaiteration: número de iteraciones

Ilustración 40 inicialización abejas

En esta ilustración se realiza la inicialización del algoritmo, como en el resto se inicia intelligence, la población de individuos y el fitness.

Se busca la mejor solución cogiendo la posición del individuo cuyo índice es el índice del menor valor almacenado en la lista de los mejores fitness y se inicializa Gbest con el valor de Pbest.

Texto

Descripción generada automáticamente

Ilustración 41División de individuos

Se dividen los individuos en varios grupos, si el número de individuos es igual o menor a 10 se dividen en 4 grupos el primero que será aproximadamente la mitad y tres grupos de 1.

Si es mayor a 10 se dividen en 4, el primero es un 10% de los individuos, el segundo son 5 individuos, el tercero que será (número de individuos - individuos en el primer grupo \* individuos en el segundo grupo – individuos en el primer grupo) / 2, y un cuarto grupo de dos individuos.

Esta división se hace para que el algoritmo pueda explorar diferentes áreas del espacio de búsqueda de manera más eficiente. Los grupos grandes tienden a centrarse más en la explotación de soluciones mientras que los más pequeños se enfocan en la exploración.

Pantalla de computadora con letras

Descripción generada automáticamente con confianza mediaSe comienza el bucle del algoritmo calculando el fitness de cada individuo y almacenándolo en una lista.

Ilustración 42 Inicio del bucle abejas

Después se ordenan de menor a mayor y se almacenan sus índices en sorted\_indices gracias a argsort de Numphy.

Best\_indices representa los índices de los individuos con los mejores valores de fitness en la población. Estos son los agentes que se consideran las fuentes de alimento más ricas, y son explotados intensamente por las abejas.

Escala de tiempo

Descripción generada automáticamenteSelected\_indices representa los índices de otro conjunto de individuos en la población. Estas no son las mejores soluciones, pero siguen siendo importantes, y se consideran como fuentes de alimento adicionales que las abejas observadoras pueden explorar.

Ilustración 43 Nuevos individuos

Se generan nuevos individuos gracias a \_\_new() y se obtiene el número de individuos nuevos.

Texto

Descripción generada automáticamente

Ilustración 44 función para nuevos individuos

Texto

Descripción generada automáticamenteEsta función genera nuevos individuos para cada individuo en l moviéndose a posiciones vecinas.

Ilustración 45 función vecinos

Genera un vecino aleatorio para un individuo who ajustando su posición a los límites.

Texto

Descripción generada automáticamente

Ilustración 46 Comprobación abejas

Captura de pantalla de computadora

Descripción generada automáticamente con confianza mediaSe comprueba si n-m es positivo antes de generar nuevos individuos y se ajustan sus posiciones a los límites de búsqueda.

Ilustración 47 Fin bucle abejas

En esta ilustración se muestra el final del bucle del algoritmo en el que se ajustan las posiciones a los límites del espacio de búsqueda de los individuos.

Se actualiza el fitness actual aplicando la función objetivo a cada individuo.

Se actualiza la mejor solución personal de cada individuo.

Se actualiza la mejor solución global si se cumple la condición.

Por último, se guarda el valor del mejor fitness y se imprime.

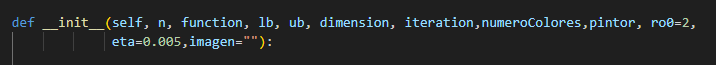
5.2.5 Ballenas

Ilustración 48 Constructor ballenas

En esta ilustración se muestra el constructor del algoritmo cuyos parámetros son:

* n: número de ballenas
* function: función a optimizar
* lb: límite inferior del espacio
* ub: límite superior del espacio
* dimension: dimensión del espacio
* iteration: número de iteraciones
* numeroColores: número de colores de la nueva imagen
* pintor: booleano que se usa para saber si pintamos imagen al final o no.
* ro0: intensidad de ultrasonido en la fuente de origen
* eta: probabilidad de distorsión de mensaje a largas distancias
* imagen: ruta de la imagen a procesar por el algoritmo

Texto

Descripción generada automáticamenteEn esta ilustración se muestra la inicialización del algoritmo de igual manera que el resto.

Ilustración 49 inicialización ballenas

Texto

Descripción generada automáticamenteSe inicializan los valores del fitness actual y del mejor fitness.

Ilustración 50 Inicio del bucle ballenas

Se inicia el bucle del algoritmo.

Una captura de pantalla de un celular

Descripción generada automáticamente con confianza mediaEn esta ilustración se muestra cómo se mueven cada uno de los individuos. Para cada individuo se busca un individuo mejor que él mismo y que sea más cercano.

Ilustración 51 movimiento de ballenas

Si se encuentra se mueve el individuo siguiendo esta ecuación:

Texto

Descripción generada automáticamente

Ilustración 52 función buscar mejor y más cercana ballena

Esta ilustración muestra cómo se busca un individuo mejor y más cercano.

Se recorren todos los individuos y para cada uno se compara el fitness del individuo desde el que se busca (u) y el fitness del individuo que toque en el bucle, si es menor se comprueba la distancia.

Se calcula la distancia entre individuos y si esta distancia es menor que la anterior que se encontró se almacena para comparar esta última con las siguientes iteraciones del bucle.

Al finalizar el bucle se devuelve el individuo con un mejor fitness que el del individuo que se evalúa y que sea más cercano.

Imagen de la pantalla de un video juego

Descripción generada automáticamente con confianza bajaEn esta ilustración se muestra cómo se halla la distancia entre individuos usando la librería de numphy.

Ilustración 53 calcular distancia entre individuos

Texto

Descripción generada automáticamentePor último, se mueven los individuos, se acotan las posiciones al espacio de solución, se actualiza el fitness actual de cada uno, se actualiza la mejor solución de cada individuo comparando el fitness actual con el mejor fitness encontrado, si el actual es mejor que el mejor almacenado se actualiza la mejor posición particular y su mejor fitness.

Ilustración 54 Fin bucle ballenas

Se actualiza la mejor solución global y se guarda el mejor fitness para imprimirlo.

# Funciones

En este apartado se documentan las funciones que se han utilizado en cada uno de los algoritmos.

## Texto Descripción generada automáticamentepintaImagen

Ilustración 55 pintaImagen

Esta última ilustración muestra como el programa pinta las imágenes resultantes del algoritmo.

Primero se obtiene la ruta del directorio actual y se sube un nivel para estar en el directorio raíz, esto es debido a que este archivo está en un nivel diferente al de la carpeta que contiene las imágenes.

Luego se accede a la carpeta de imágenes.

Se forma el nombre a la imagen resultante que se va a almacenar en formato ALGORITMO\_NUMCOLORES\_IMAGEN.

Se guarda la imagen cuantizada, se hace una comprobación para en caso de que no se haya podido guardar de error y no intente leer la imagen.

Después se leen las dos imágenes, la original y la cuantizada.

La comprobación de si es un jpg la redimensiona es porque al hacer las pruebas y pasarle una imagen personal hecha con un teléfono, el programa me pintaba las imágenes en un formato muy grande, se salía de la pantalla.

Si se el parámetro de pintar la imagen se ha pasado a true pinta las dos imágenes, después el programa espera a que se pulse cualquier tecla, al pulsarla se destruyen las ventanas y termina.

Este método se llama al final de cada algoritmo.

## Texto Descripción generada automáticamentepreparaImagen

Ilustración 56 preparaImagen

Esta ilustración muestra el método que se llama para preparar las imágenes para operar con ellas, se encarga de leer la imagen, redimensionarla si es jpg y redimensionar la imagen a una matriz 2D de pixeles y luego se devuelve la matriz junto con la imagen.

## Texto Descripción generada automáticamentegeneraCuantizada

Ilustración 57 generaCuantizada

Esta ilustración muestra el método encargado de generar una imagen cuantizada.

Primero llama a preparaImagen para preparar la imagen para su uso.

Se eliminan los pixeles duplicados, esa línea es debida a que a raíz de actualizar la librería sklearn se experimentaron errores que se corrigieron con esto.

Se ajustan el número de clústeres al número de pixeles únicos si es menor que tam\_paleta que es el tamaño de la paleta que se pide o el número de colores.

Se aplica el Kmeans de la librería sklearn a los pixeles únicos para encontrar los clústeres.

Se guardan los valores de las etiquetas de cada pixel y se obtienen los centroides para luego reemplazar cada pixel en la imagen original por el color del clúster al que pertenece.

Por último, se devuelve la imagen cuantizada.

## getMse

Pantalla de computadora con letras

Descripción generada automáticamente con confianza mediaEn esta ilustración se muestra cómo se calcula el mse, se prepara la imagen, se genera la cuantizada y se llama a mean\_mean\_squared\_error de la librería sklearn para calcular el error cuadrático medio

Ilustración 58 función MSE

## getMae

Captura de pantalla con la imagen de una pantalla

Descripción generada automáticamenteEn esta ilustración se muestra cómo se calcula el mae, se prepara la imagen, se genera la imagen cuantizada, se redimensiona y se calcula el mae gracias a numphy. Devuelve el error absoluto medio.

Ilustración 59 getMae

## getSsim

Texto

Descripción generada automáticamente

Ilustración 60 getSsim

En esta ilustración se muestra cómo se calcula el ssim, igual que en las anteriores se prepara la imagen y se genera la imagen cuantizada, luego se redimensiona la imagen a una matriz de dos dimensiones para luego calcular el ssim gracias a la librería skimage

Como esta librería devuelve el índice de similitud en el que 0 no se parecen y 1 si se parecen, invierto el valor al restar 1 menos lo que devuelve ssim para que se ajuste a los algoritmos.

## getMssim

Texto

Descripción generada automáticamente

Ilustración 61 getMssim

Esta función calcula el mssim de la misma manera que el ssim, la diferencia son los parámetros que se mandan a ssim lo que en vez de devolver el índice de similitud estructural (ssim) devuelve el índice de similitud multi escalar (mssim).

Como en la función anterior se invierte el índice para ajustarse a los algoritmos.