Universidad de Guadalajara

Centro Universitario de Ciencias Exactas e Ingenierías CUCEI

Ingeniería en Computación

Materia: Seminario de Solución de Problemas de Inteligencia Artificial

Sección: D04

Profesor: Hernández Barragán, José de Jesús

Alumno: Geovanni Quezada Torres

Código: 212753578

Proyecto Final

Proyecto Final

Elección:

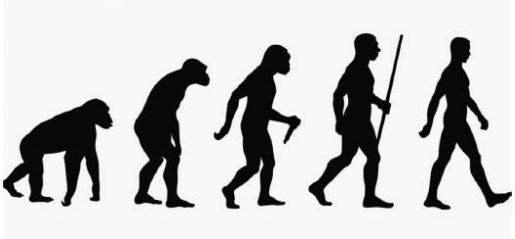
Evolución Diferencial

¿Porqué?

Mi elección fue Evolución Diferencial, ya que es muy sencilla de programar y de implementar, además sabiendo aplicarle una restricción resulta ser un algoritmo muy eficiente.

Información sobre Evolución Diferencial

El algoritmo de Evolución Diferencial (DE) es un algoritmo evolutivo de optimización, inspirado en la inteligencia colectiva en el proceso de evolución.



DE se caracteriza por tener una población de individuos y utilizar las operaciones de mutación, recombinación y selección como mecanismo para mejorar sus individuos en cada generación.

El algoritmo comienza con la inicialización aleatoria de la posición ${}^{t}x_i$ de los individuos i = 1, 2, 3, · · · ,N en la generación t donde N es el total de individuos en la población. Cada individuo xi representa una solución potencial.

Por cada individuo se crea un vector mutado utilizando:

$$ {}^{t}v_i = {}^{t}x_{r1} + F\,({}^{t}x_{r2} - {}^{t}x_{r3}) $$

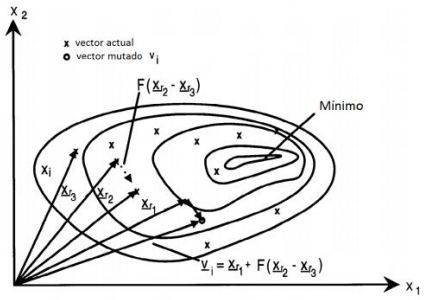
donde:

r1, r2, r3 2 [1,N], son números aleatorios tal que r1 6= r2 6= r3 6= i

vi es el vector mutado

F 2 [0, 2] es un factor que controla la amplificación de la diferencia (xr2 − xr3)

La siguiente imagen muestra el proceso de mutación del algoritmo DE.



M.

Después, $^{t}x_i$ es mezclado con $^{t}v_i$ bajo el siguiente esquema de recombinación:

$$u_{ij} = \begin{cases} v_{ij} & \text{si } r_a \leq C_R \\ x_{ij} & \text{cualquier otro caso} \end{cases}$$

donde:

j = 1, 2, 3, · · · ,D donde D es la dimensión del problema

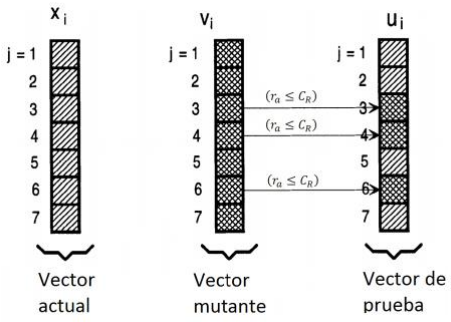
r_a 2 [0, 1] es un numero aleatorio

u_i es el vector de prueba

CR 2 [0, 1] es una constante de recombinación

La siguiente imagen muestra el proceso de recombinación del algoritmo DE para un

problema de dimensión D = 7.



Finalmente, se realiza la operación de selección como sigue

$$ {}^{t+1}\mathbf{x}_i = \begin{cases} {}^{t}\mathbf{u}_i & \text{si } f\left({}^{t}\mathbf{u}_i\right) < f\left({}^{t}\mathbf{x}_i\right) \\ {}^{t}\mathbf{x}_i & \text{cualquier otro caso} \end{cases} $$

Notas:

El factor de aplicación F, establece el rango de diferenciación entre (xr2 − xr3) con el objetivo de evitar estancamiento en el proceso de búsqueda.

La constante de recombinación CR modifica la relación entre exploración y explotación.

Algoritmo

$F, C_R \leftarrow$ definir parámetros
$\mathbf{x}_i \leftarrow$ inicializar $i \in [1, N]$ individuos aleatoriamente tal que $\mathbf{x}_i \in \mathbb{R}^D$
Hacer
    Desde $i = 1$ **Hasta** N
        $\mathbf{v}_i \leftarrow \mathbf{x}_{r_1} + F\left(\mathbf{x}_{r_2} - \mathbf{x}_{r_3}\right)$
        Desde $j = 1$ **Hasta** D
            Si $r_a \leq C_R$
                $u_{ij} \leftarrow v_{ij}$
            Si No
                $u_{ij} \leftarrow x_{ij}$
            Fin Si
        Fin Desde
        Si $f\left(\mathbf{u}_i\right) < f\left(\mathbf{x}_i\right)$
            $\mathbf{x}_i \leftarrow \mathbf{u}_i$
        Fin Si
    Fin Desde
Mientras que se cumpla el total de generaciones G

Procesamiento de imágenes

La detección de objetos es una tarea importante ya puede ser utilizada para en diferentes áreas de investigación como robótica, medicina, visión artificial, etc.

Función Objetivo NCC

Correlación Cruzada Normalizada (NCC) es un método que se utiliza para medir la similitud entre dos imágenes.

Normalmente se realizan tareas de detección de plantillas (sub-imagen) dentro de una imagen dada.

Utilice el método NCC de comparación de plantillas por las siguientes razones:

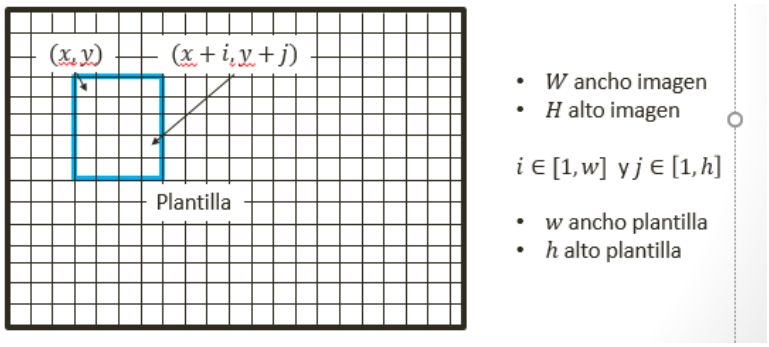
Las ventajas del método NCC sobre otros métodos de “Comparación de plantillas” son:

-El método NCC utiliza una simple ecuación para medir el grado de similitud y es fácil de programar.

-El método NCC es robusto a cambios en el brillo y contraste de las imágenes comparar.

La desventaja es que es invariante a la rotación y translación.

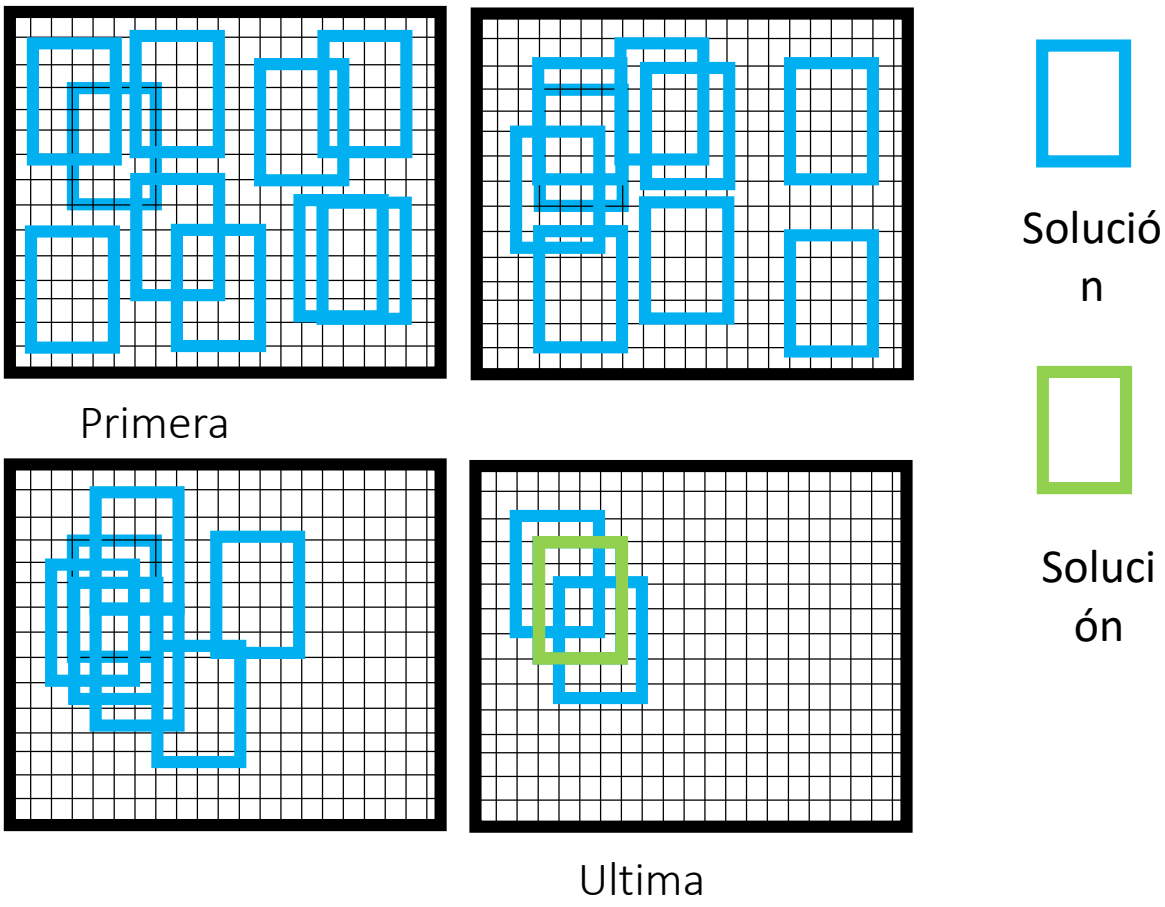
El método tradicional de NCC consiste en desplazar la plantilla sobre la imagen en incrementos de $x \in [1, W]$ e $y \in [1, H]$, y medir en cada desplazo el grado de similitud.



Se utilizó el algoritmo de Evolución Diferencial para resolver el problema de detección de plantillas (TM).

- Cada individuo en la población calculará la medida de similitud utilizando la función NCC como función objetivo.
- La propuesta evita el desplazo sobre la Imagen, lo que reduce tiempo computacional.

Ejemplo de solución:



Para resolver el problema de comparación de plantillas, se propone maximizar una función objetivo f como sigue,

$\max f(\mathbf{p})$ sujeto a $\mathbf{p}_l < \mathbf{p} < \mathbf{p}_u$

$$\mathbf{p} = \begin{bmatrix} x \\ y \end{bmatrix}$$

El vector $\mathbf{p}$ representa la posición del "desplazamiento de la plantilla" sobre la imagen.

Los vectores $\mathbf{p}_l$ y $\mathbf{p}_u$ representan los limites en la imagen (el tamaño en pixeles de lo ancho y alto).

La función objetivo f es igual a la función NCC, es decir cada individuo calcula el fitness evaluando el NCC en su posición.

Se propone penalizar aquellos individuos que se salgan del espacio de trabajo como sigue,

$$p_{ij} = \begin{cases} p_{ij} & \text{si } p_{lj} < p_{ij} \text{ y } p_{ij} < p_{uj} \\ \text{recalculado} & \text{otro caso} \end{cases}$$

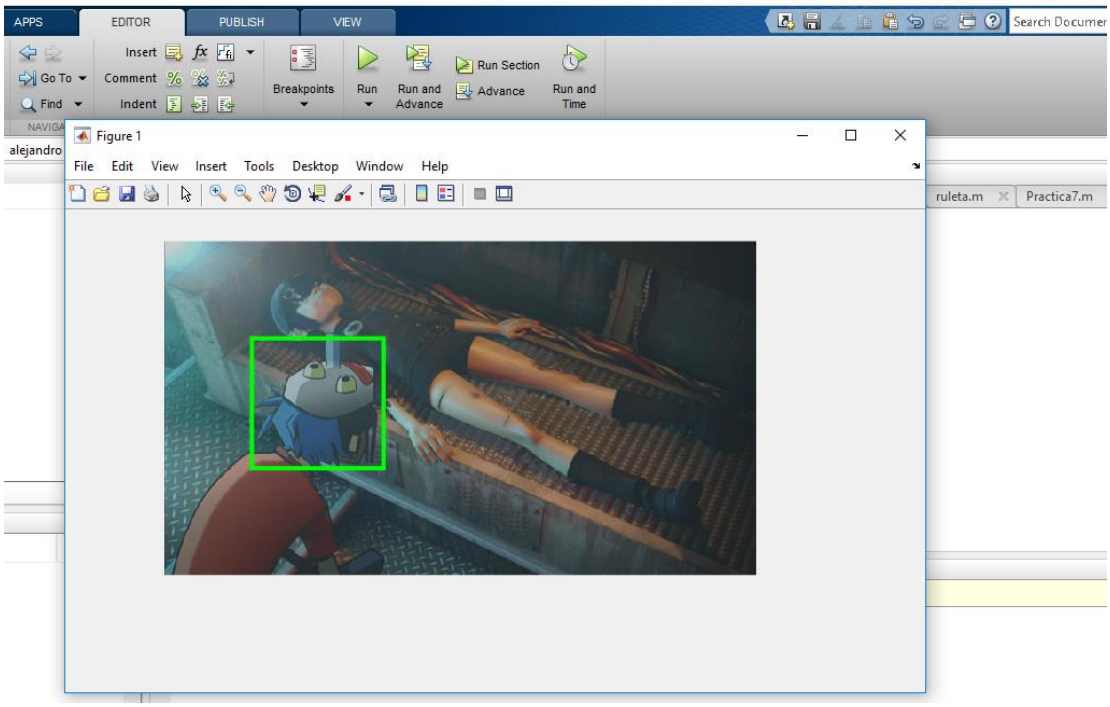
donde i representa al individuo y j la dimensión.

Resultados:

Imagen Principal:

Plantilla:

Resultado al ejecutar Evolución Diferencial con restricción:



Conclusión:

Utilicé el algoritmo de Evolución Diferencial, ya que me fue el más fácil para programar, además de que da resultados óptimos aplicándole la restricción, porque tuve que aplicar restricciones en la implementación de la detección de plantillas en imágenes, porque si se salía de los límites de búsqueda, el programa tronaba, así que fue necesario agregarle una función llamada "penalización".

Lo que hace ésta función es que si algún individuo en alguna de las iteraciones se sale del rango de la imagen lo recalcula de nuevo dentro del espacio de búsqueda, esto hace que tenga probabilidad de quedar más cerca del máximo global y así se explote esa solución.

En el algoritmo tuve que cambiar algunas cosas, porque está diseñado para encontrar el mínimo global de una función, lo único que tuve que cambiar fue a la hora de la etapa de la selección, cambiar el signo que compara la función

objetico con el vector de prueba, cambiarlo a que, si es mayor el vector de prueba, se queda con éste, si no, sigue con los valores de la función objetivo.

Otra cosa que tuve que cambiar, fue para elegir el mejor valor, ya que en este caso buscamos el valor más grande, fue la variable “best” igualarla a “-inf” porque en éste caso se busca maximizar, entonces se compara con un numero negativo.

Y finalmente seleccionamos nuestro mejor valor para mostrarlo.

Lo más complicado para mí, fue entender la lógica para aplicar la función NCC, en vez de alguna función de prueba con las que estuvimos trabajando todo el semestre.