Algoritmos Genéticos

Inteligencia Artificial. Concept Art: Personajes, Escenarios y Props

Jorge Bárcena Lumbreras
Miguel Ángel Gil Martín
02/02/2020

Índice

Descripción de la herramienta realizada

La práctica desarrollada consiste en, mediante un algoritmo genético, generar una copia de la foto que nosotros introducimos, mediante figuras circulares. La copia generada por el algoritmo será del mismo tamaño que la copia original, e intentará adaptarse lo mas posible a la imagen de referencia.

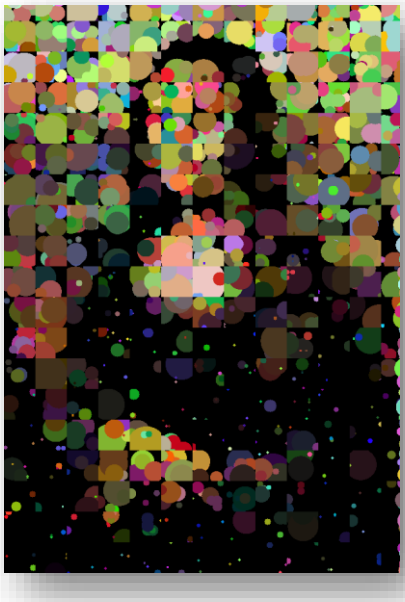
Análisis de resultados

Hemos realizado una serie de pruebas, cambiando las variables del algoritmo y hemos obtenido gran variedad de resultados. Las variables que podríamos cambiar eran el tamaño de los chunk, el tiempo que estará el algoritmo en cada chunk, la cantidad de población, y la cantidad de esferas que tiene cada uno de los individuos. Estos han sido los resultados:

Chunk de 50x50 – 15 segundos por chunk – 20 Individuos – 10 esferas / individuo

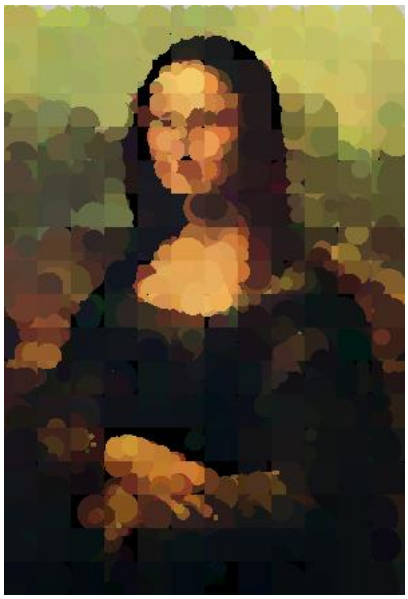
En esta configuración podemos ver como el algoritmo esta bastante cerca de la imagen final, tenemos que tener en cuenta la resolución de la imagen original que es de 300 x 300, por lo que chunks de 50 x 50 no son demasiado pequeños y la calidad de la imagen final se ver afectada.

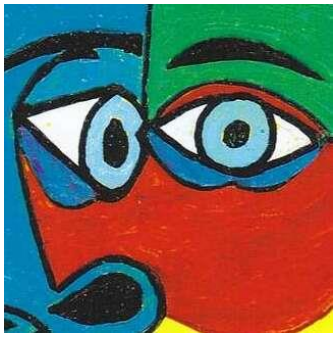
Chunk de 50x50 – 2 segundos por chunk – 20 Individuos – 10 esferas / individuo

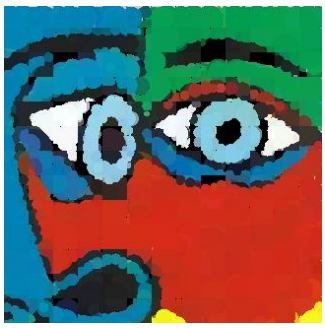


Afectada Con este formato de configuración podemos ver como el que esté dos segundos por chunk afecta de manera directa a la calidad final de la imagen, aun así podemos ver como se ha marcado la silueta de manera correcta y el algoritmo se intenta parecer a la imagen.

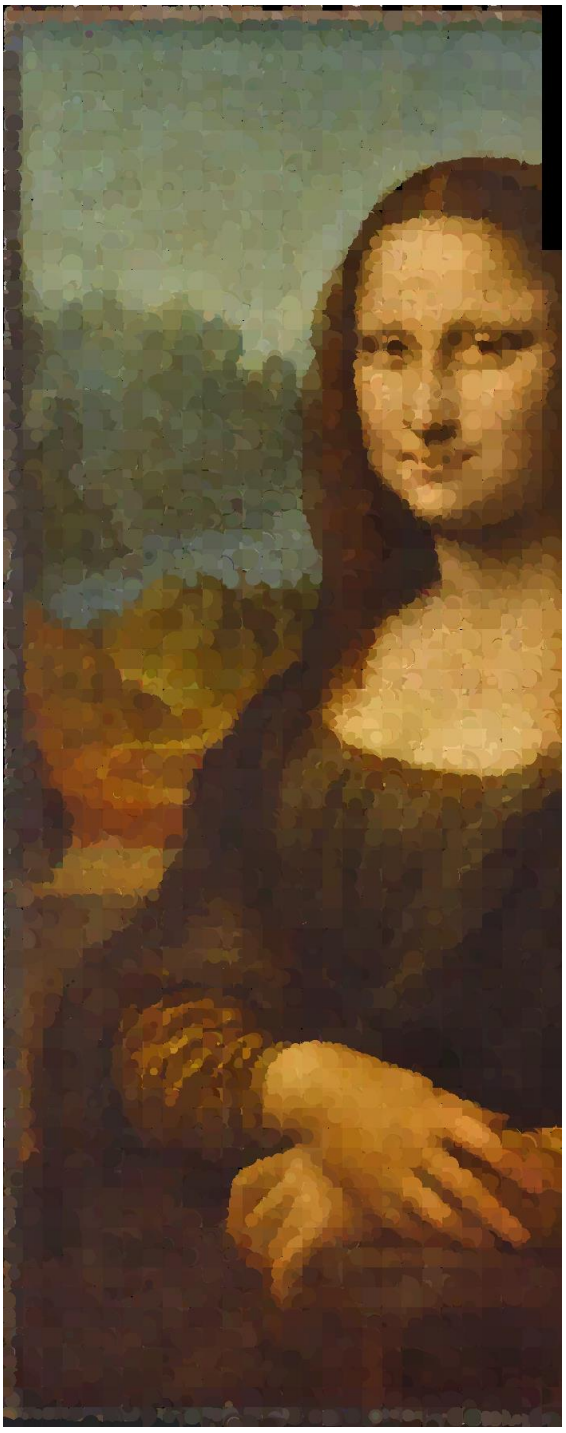
Chunk de 25x25 – 8 segundos por chunk – 20 Individuos – 10 esferas / individuo







Tras hacer varias pruebas hemos obtenido que está es la configuración mas optima para ejecutar el algoritmo con satisfacción es esta configuración, podemos ver como el resultado final es bastante fiel a la realidad. También depende mucho de la calidad de la imagen original, cuanta más calidad, más tardará en generarse la imagen final. Estas imágenes han tardado entre y 20 minutos en generarse, y el resultado es bastante fiel. A continuación, se presenta una réplica de la monalisa, pero con un tamaño de imagen original de 1000 x 1000 pixeles, es resultado de esta imagen final es bastante bueno, pero el algoritmo ha tardado 2 horas en generar la parte visible de la imagen:



Estructura de clases

Gen

En esta clase se encuentran los datos que se heredan y mutan, que son, la posición del centro de la circunferencia, el color y la posición de zetas. También contiene funciones para acceder y modificar todos estos valores

Genetic Element

Es una clase abstracta que contiene a Gen, también contiene dos funciones abstractas para inicializar y otra para pintar.

Genetic Individual

Para mejorar la eficiencia del algoritmo no solamente tomamos las esferas como individuos de nuestro algoritmo, sino que, a su vez, existe esta clase que contiene una lista de esferas para pintarlo. De esta manera no comprobamos esfera por esfera si tiene un score mayor o menor, si no que simplemente comprobamos todo el objeto de genetic individual, y este es el que mutamos, a través de mutar las propias esferas.

Genetic Parents

Esta clase funciona de la misma forma que lo hace el genetic Individual, solo que en el momento de mutar, coge mutaciones de varios genetic individuals, para acercarse a la solución más rápido.

Genetic Controller

En esta clase como indica su nombre se controla la herencia de las próximas generaciones que se generan.
Esta clase es un Singleton y contiene información como el número de círculos que se generan por chunk, dimensiones del chunk, tiempo que se invierte por chunk, el ratio de mutación que existe por generación, etc.
Contiene las funciones que mandan generar una nueva generación y guarda las puntuaciones de parecido de cada generación, se podría decir que el Algoritmo completo converge en esta clase.

Image Comparator

Aquí se encuentra la función de comparar texturas, que básicamente retorna cuanto de parecido es una lista de píxeles respecto de otra. Este parecido se evalua de la siguiente forma:

Cogemos el mismo pixel de la imagen de referencia y la imagen generada, y comprobamos los componentes de color de ese pixel. Existe una variable error que lleva un conteo del error general de la textura, cuando el componente del color no es igual, se suma el valor absoluto de la diferencia del pixel 1 con el pixel 2.

Image Manager

Contiene todas las mallas e imágenes de la práctica, se encarga de gestionar los chunks y de actualizar las texturas.

Sphere

Esta clase hereda de Genetic Element, esta clase se encarga de almacenar su gen, gestionar su inicialización y de pintarse en la textura que le corresponde.

Futuras mejoras

Como futuras mejoras se podría incluir, mas primitivitas como elipses, cuadrados, o polígonos. También podríamos incluir un componente de Alpha para que las esferas o primitivas tengan cierta variación en función de donde estén.

Problemas Encontrados

Al principio teníamos dos texturas, en una se encontraba la imagen inicial y en la segunda se iba generando nuestra nueva imagen, esto hacía que fuese muy complicado que las células mutaran para bien, para solucionar esto, pusimos una comprobación global de cuánto había mejorado la imagen y añadimos una textura nueva, en la nueva textura se va ejecutando el algoritmo y cuando la imagen mejoraba su símil con la original se copiaba la textura y se pasaba a la textura final, cuando el algoritmo determinaba que el símil con la original bajaba copiaba la textura que más se acerca al original y vuelve a probar. Esto hizo que fuese muchísimo más rápido.

Otro problema que veíamos es que el algoritmo solo se desenvolvía bien cuando las imágenes eran de un tamaño entorno a 50 x 50 píxeles, para poder pasar por el algoritmo imágenes más grandes que estas dimensiones, pusimos que fuese generando por chunks, de esta forma seguía trabajando con texturas de 50 X 50 pero reiteradamente.

También para comprobar la distancia que existía entre las coordenadas rgb de la imagen original a el color de nuestros círculos intentamos usar la distancia de Mahalanobis, pero al tener que hacer cálculos con medias ponderadas por cada canal, se creaba una serie de bucles que terminaba perjudicando la velocidad del algoritmo y terminamos quitándolo.