FRACTALES Y PROBABILIMO: REPRESENTACION DE ELEMENTOS NATURALES

Vicente Martínez Franco

Algoritmos y computacion Curso 2015/2016

DESCRIPCION DEL PROBLEMA:

En este trabajo he realizado un pequeño estudio de el uso de la probabilistica en el dibujado de fractales mediante algoritmos en un ordenador. Aunque en un principio puedan parecer temas separados, ya que los fractales suelen realizar siempre la misma transformacion en cada paso, en cuanto intentamos utilizarlos para representar elementos presentes en la naturleza (uno de sus usos mas frecuentes, y quizas el mas interesante) esto ya no vale y debe empezarse a utilizar un cierto grado de aleatoriedad.

Para quien no sepa que son los fractales estos son figuras geometricas que se repiten dentro de si mismas a una cierta escala, es decir, que al coger una parte del fractal esta sera, al menos aproximadamente, una copia de la original.

Para crear un fractal se necesitan 2 elementos: un iniciador, que es el elemento con el que se empieza, y un generador, que se aplica sobre el elemento original y posteriormente sobre cada una de las zonas que tengan la misma forma que la original.

El interes de usar aleatoriedad es evitar que el resultado obtenido sea demasiado regular, ya que elementos como estos no se suelen encontrar en la naturaleza. Sin embargo si realizaramos la decision de que va a ocurrir de forma completamente aleatoria el resultado seria completamente impredecible y en muchos casos podrian salir cosas distintas a las esperadas.

Par evitar esto podemos introducir una serie de reglas a la aeatoriedad, tantas como queramos, de forma que le resultado se ajuste a una serie de parametros.

EJEMPLO CONCRETO: ALGORITMO DEL PUNTO MEDIO

El ejemplo concreto que voy a tratar es el de el algoritmo del punto medio de la generacion de terrenos.

Lo que hace este algoritmo es tomar un cuadrado (o rectangulo) como elemento iniciador. Dentro de este cuadrado calcula el punto medio y la altura de este como la media de alturas de las esquinas. Posteriormente aumenta o decrece la altura de este punto. Despues divide el cuadrado en otros 4 y aplica el algoritmo recursivamente en ellos.

El elemento de aleatoriedad lo introduciriamos en la variacion de altura del punto medio. Las restriciones pueden aplicarse sobre zonas, alturas, numero de iteracion, etc. Por ejemplo para conseguir una pendiente en una zona solo tendriamos que , por ejemplo, definir la altura en lo bordes de la pediente y calcular la atura de los puntos en vez de como la media de las esquinas del cuadrado según su posicion en el plano.

PSEUDOCODIGO:

```
algPuntoMedio(Punto arrDe,abDe,arrIz,abIz) {

Punto puntoMedio,puntoDerecho,puntoIzquierda,puntoArriba,puntoAbajo;

        puntoMedio.heigth=(arrDe.heigth+abDe.heigth+…)/4;

        puntoMedio.x=(arrDe.x+arrIz.x)/2;
```

```
        puntoMedio.z=(arrDe.z+abDe.z)/2;

        puntoDerecho.x=arrDe.x;

        puntoDerecho.z=puntoMedio.z;

        puntoDerecho.heigth=(arrDe.heigth+abDE.heigth)/2;

        .

        .

        .

        puntoMedio.heigth+=rand();

        algPuntoMedio(arrDE,puntoDerecho,puntoArriba,puntoMedio);


        .


        .

    }
```

COSTE:

El coste de este algoritmo es de 4 elevado a numero de iteraciones -1, es decir un orden exponencial. Sin embargo una simple iteracion produce una gran mejora en el resultado luego el coste seria asumible, sobre todo teniendo en cuenta el coste en tiempo de los procesos de iformatica grafica y que no deberia ser aplicado muchas veces. Ademas la utilida de este algoritmo es evitar que una persona deba diseñarlo a mano lo cual supondria una mayor cantidad de tiempo y podria no producir tan buenos resultados.