ALGORITMIA BÁSICA

Práctica 2 - Programación dinámica

Nicolás de Rivas Morillo (843740)
Cristina Embid Martínez (842102)

Resumen

En esta práctica se ha llevado a cabo el diseño e implementación de un algoritmo que a partir de una imagen reduce su anchura. Para decidir qué píxeles recortar se define la noción de costura. Las costuras son secuencias de píxeles que recorren la imagen de un extremo a otro, donde cada píxel está a lo sumo separado por el siguiente en diagonal. Además la costura que tomamos para eliminar es la que minimiza la función *C* definida en el enunciado basada en el coste de energías basados en brillos de los píxeles.

Implementación

En primer lugar, se ha creado un matriz con bordes adicionales cuyo brillo es cero para tratar los bordes de la imagen original. Para encontrar la costura de menor energía se recorre la matriz de energía fila por fila, acumulando la energía mínima hasta ese pixel teniendo en cuenta la posición de dicho píxel (dependiendo si se trata de un pixel en los bordes o en el centro). Al final partiendo de la posición de menor energía en la última fila, se reconstruye siguiendo los predecesores almacenados.
Además cuando se trata de varias iteraciones seguidas se evita el cálculo de todo nuevamente, reduciendo considerablemente los tiempos de cálculo.

Análisis de las pruebas realizadas

Las pruebas se han realizado en imagenes con formato *.bmp* o *bitmaps* por dos motivos principales, por un lado se tratan de imágenes sin compresión lo cual nos ayuda a simplificar toda la operativa y descartar errores externos al programa, por otro lado usan codificación RGB la cual es la más cómoda teniendo en cuenta como se ha definido el brillo de cada pixel, si hubiésemos elegido otro formato que usase codificación de colores CMYK, RGBa, HSL o otros hubiésemos tenido que hacer una transformación del formato. Como en la anterior práctica hemos intentado desarrollar unas pruebas de dificultad paulatinamente ascendentes y que se enfocan en diferentes aspectos.

Prueba nº1: *Número de columnas que se reduce la imagen = 1*

La prueba más sencilla posible y fácil de verificar. Una serie de línear verticales de un píxel de grosor alternando blanco y negro. En este caso observamos que elimina la primera línea blanca, lo cual concuerda con el algoritmo, los píxeles negros tienen brillo 0 y por lo tanto todos los píxeles blancos (que están rodeados por negros) tienen energía 0 lo cual minimiza la suma de energía en la costura. Coger la línea izquierda es simplemente por cómo tomamos el mínimo.

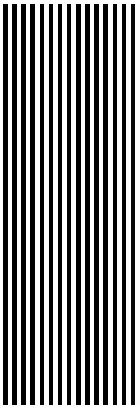

Imagen original

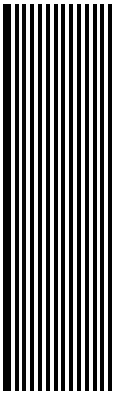

Imagen final

Prueba nº2: *Número de columnas que se reduce la imagen = 150*
Se trata de una imagen compuesta de figuras geométricas por lo que es fácil ver si se han reducido las dimensiones. Vemos en esta prueba que este algoritmo es bueno en el sentido de que mantiene la continuidad de las formas, por ejemplo, el círculo de la imagen no se parte en dos ni se corta si no que pasa a ser un óvalo.

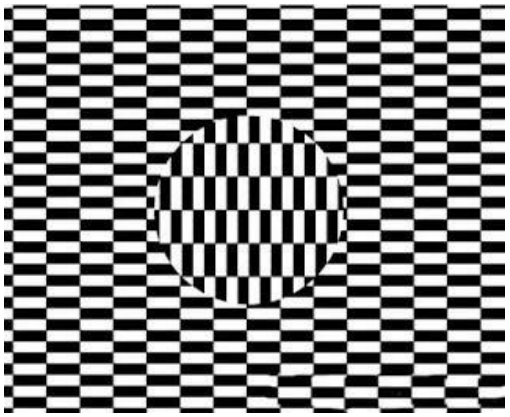

Imagen inicial

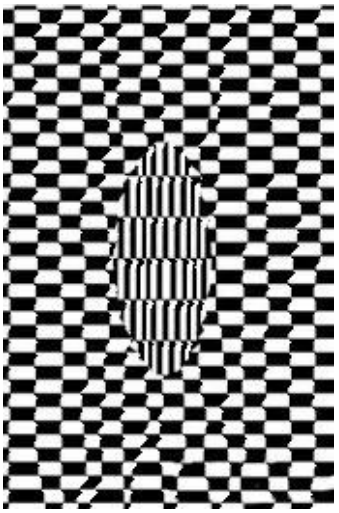

Imagen final

Prueba nº3: *Número de columnas que se reduce la imagen = 50*
Similar a la prueba anterior pero con formas más irregulares, podemos hacer la misma observación.

Prueba nº4: *Número de columnas que se reduce la imagen = 30*
De nuevo líneas, como en la primera prueba, pero en este caso en diagonal.

Prueba nº5: *Número de columnas que se reduce la imagen = 60*
En la siguiente imagen se aprecia como se ha reducido el espacio entre las letras inclinadas sin causar una deformación notable.

Prueba nº6: *Número de columnas que se reduce la imagen = 60*
En la siguiente imagen se muestra la señal de Stop que mantiene las letras pero no pasa lo mismo con la forma ya que se ve distorsionada.

Prueba nº7: *Número de columnas que se reduce la imagen = 180*
En la siguiente imagen se aprecia la deformación de algunas de las latas ya que se ven más estrechas y comprimidas.

Imagen inicial

Imagen final

Prueba nº8: *Número de columnas que se reduce la imagen = 190*
Se trata de un paisaje en el que podemos ver que se conservan las formas generales del escenario, aunque se percibe que el puente es más estrecho haciendo que cambie la sensación de profundidad y perspectiva.

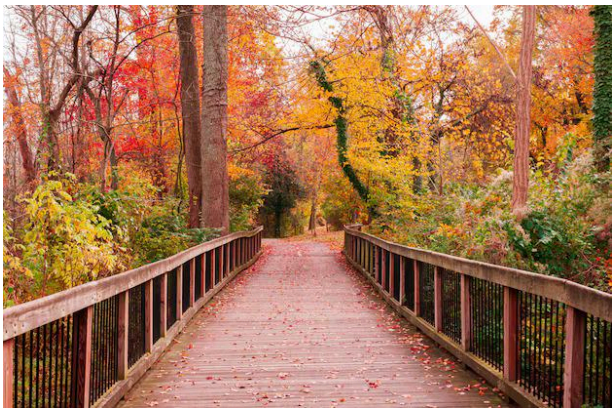

Imagen inicial

Imagen final

Prueba nº9: *Número de columnas que se reduce la imagen = 210*
En la imagen final, se puede ver la alteración de la forma de la esfera y el cambio de proporción de la imagen.

Análisis de tiempos y eficiencia

Para analizar tiempos vamos a usar la librería *timer* de python y el decorador con ese mismo nombre, esto además nos permitirá analizar los tiempos función por función. El decorador *@timer* está aplicado tanto en el constructor de nuestro objeto que calcula la costura inicial como en la función que recalcula la siguiente costura a partir de la anterior.
Tras la ejecución, se almacena en el fichero *tiempos.log* los tiempos de cada una de las pruebas realizadas.
Observamos así el cambio entre el tiempo de computación de la primera iteración comparada con el resto, estas son aproximadamente un 500% más rápidas. La primera iteración tiene que computar (En una imagen de dimensiones $m \times n$, suponiendo $m > n$) $m \cdot n$ brillos, $m \cdot n$ energías y $m \cdot n$ ejecuciones de la función *C*, sin embargo, en el peor caso una de las siguiente iteraciones calcula: 0 brillos, $4n$ energías y $4n + n^2$ ejecuciones de la función *C*. Aún se pueden afinar algunos casos pero no creemos que su valía sea tanta como los ya implementados, tanto en tiempo invertido para implementarlos, como en legibilidad del código y tiempo de ejecución.