Autores: Luis Francisco Blanco Cadavid
Alex Carballo Henriques

Identificación de Texto

Introducción:

Lo primero que hicimos fue identificar el problema propuesto. Lo que se nos planteaba principalmente era como reconocer el texto entre dos imágenes: El texto y una plantilla que contiene los diferentes caracteres que lo componen.

Visión Global:

Como lo que se nos pide es enseñar el texto por consola en primer lugar nos planteábamos hacer un recorrido por el código ASCII utilizando o una variable carácter, sin embargo el código extendido no era reconocido correctamente por las librerías de c. Por lo que optamos por la opción de crear un array de caracteres y como cada letra esta por orden en la imagen usada como plantilla, dentro del "for" al ir avanzando por los cuadrantes de la imagen plantilla mostramos el carácter por consola accediendo a la misma posición del array.

Para identificar el texto avanzamos en cuadrantes de 15*35 (lo ocupado por cada carácter) tanto en el texto como en la plantilla. Cada cuadrante del texto recorremos los cuadrantes de la plantilla identificando la suma de las diferencias entre los píxel de cada cuadrante. Si la diferencia es menor a un determinado valor, identificamos esa letra y la mostramos por pantalla. Empleamos un umbral a la hora de calcular las diferencias para tener en cuenta el error al comparar.

Para establecer los distintos cuadrantes que ocupan los caracteres en la plantilla tuvimos en cuenta lo indicado en la especificación del trabajo, mientras que el texto realizamos una búsqueda intensiva pixel a pixel comprobando la suma de la diferencia de los pixeles del tamaño de un cuadrante. Es decir avanzamos el cuadrante por la imagen para encontrar el carácter.

Para amenizar la interfaz gráfica de la aplicación añadimos a la imagen texto un recuadro que rodea el cuadrante en el que se está realizando la comparación entre caracteres. En cada avance refrescamos la imagen y el cuadro va avanzando según se

Autores: Luis Francisco Blanco Cadavid
Alex Carballo Henriques

Generación array:

char letra[97]={'!','"','#','$','%','&','Â

','.','/','0','1','2','3','4','5','6','7','8','9',':',';','<','=','>','?',','

A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T

','U','V','W','X','Y','Z','[','/',']','^','`','a','b','c','d','e','f','g','h'

,'i','j','k','l','m','n','o','p','q','r','s','t','u','v','w','x','y','z','{',

'|','}','~','a','e','i','o','u'};

Accediendo a una posición del array mostramos el carácter de esa posición.

Cálculo de diferencia entre cuadrantes:

for(fila=0;fila<=35;fila++)

//Inicializamos los punteros de las imágenes.

//Img1=Imagen Texto

//Img2=Imagen Plantilla

//Img3=Clonación Img1. Utilizado para refrescar la imagen

for(columna=0;columna<=15;columna++)

*dP=abs((int) (*pImg1 - *pImg2));*

if(fila==0||columna==0||fila==34||columna==14)

} //Repetimos esta operación 3 veces. Una por cada //componente del pixel.

Si la suma de las diferencias en el cuadrante es menor a un determinado valor será entonces cuando mostramos la letra por pantalla.

Problemática:

Debido a los problemas en instalar el compilador e intentar el correcto funcionamiento de las librerías, el mayor gasto de tiempo se produjo sobre la resolución de dicho problema. Optamos por la solución de utilizar la máquina virtual del aula de prácticas así como de utilizar las librerías de OpenCV de la versión 2.2 debido a que no nos correspondía el tratamiento de video. A pesar de esto, la ejecución de la aplicación se realizó de forma correcta. Las pruebas dieron a entender que el programa atendía a lo solicitado en el enunciado del proyecto.

Autores: Luis Francisco Blanco Cadavid
Alex Carballo Henriques

Segunda Entrega:

La segunda entrega consta del uso de la librería SSE2 y/o hilos adaptados al proyecto anterior. La decisión que se ha tomado es el uso de paralelismo integrando la librería SSE2 lo que ha mejorado ligeramente el rendimiento del programa, para realizar la comparación se ha usado el método "sad" de dicha librería que compara dos m128i, es decir, 16 bytes de dos imágenes distintas y nos proporciona la diferencia.

Comparar Imágenes:

```
//Compara dos imagenes usando la libreria SSE2.
int compararImagen(__m128i *pImg1,__m128i  *pImg2)
{
   __m128i pmImg = _mm_loadu_si128 (pImg1);
   __m128i pi = _mm_loadu_si128 (pImg2);
   __m128i t = _mm_sad_epu8(pmImg, pi);
   int *ttt = (int*) &t;

   return ttt[0]+ttt[2];
}
```

En este fragmento de código compararemos dos bloques de imágenes de m128i (16 bytes) para ello esta función tendrá como argumentos dos punteros que se inicializaran uno al fragmento de texto en el que estamos buscando y el otro a la letra del patrón que queremos comprobar si existe en el texto.

Los bucles para completar la comparación del fragmento de 16x35 pixeles se encontrara en otra función, recorrer imagen, que se encargará de repetir esta comparación tres veces para abarcar las tres componentes de color y las 35 filas del fragmento.

Problemática:

Debido a que nuestro proyecto originalmente era de 15X35pixels pero a la hora de comparar al ser el registro m128i de 16 bytes tomaba un pixel más de la imagen para comparar, al principio intentamos hacer un set de ese pixel pero debido a que ese pixel en concreto siempre se va a encontrar en blanco debido a la separación de las letras tanto en el texto como en el patrón, permitimos que compara sin ningún problema a ese pixel a mayores dando exactamente el mismo resultado.

Por otro lado también se intentó añadir theads al proyecto pero después de realizar varias pruebas con ellos este daba diversos fallos que debido a nuestra falta de tiempo fue imposible corregir, lo que resulto en que no usásemos hilos en este proyecto